

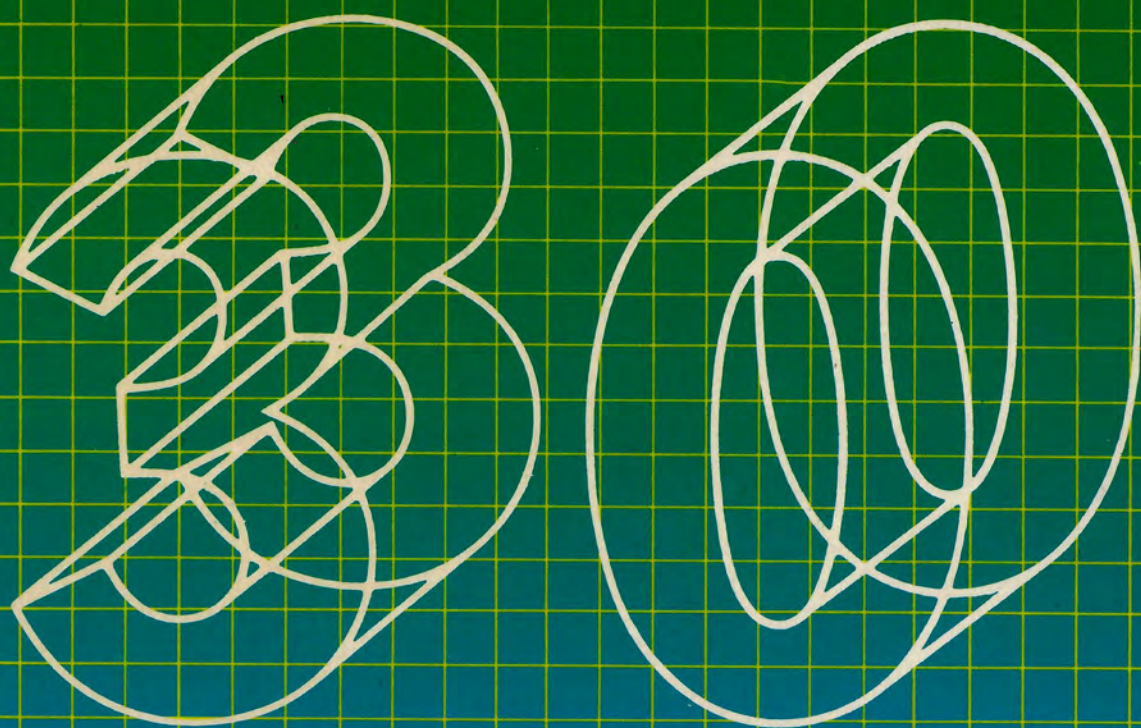
BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,

IBM, SPECTRUM, COMMODORE Y MSX



BRICOLAGE INFORMATICO

MANEJO DE ELEMENTOS GRAFICOS

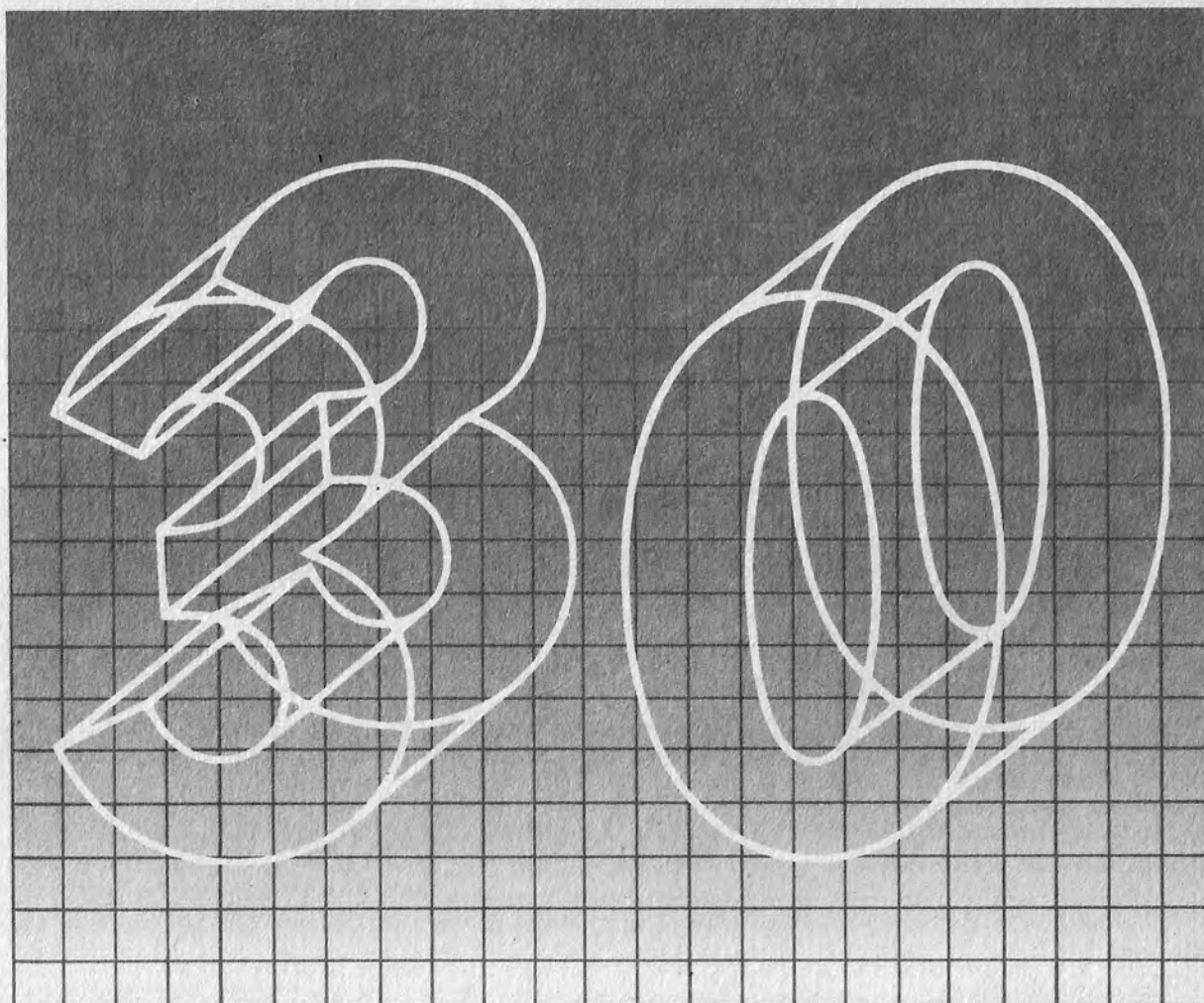
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

TALLER DE INFORMATICA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 30. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMÁTICA APLICADA: Jesús Bocho, Licenciado en Informática; Alejandro Marcos, Licenciado en Química. «El Taller de Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-152-5

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

GRAFICAS EMA. (Madrid)

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Agosto, 1987

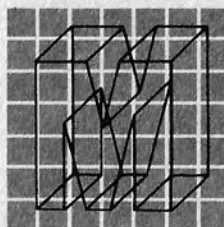
P.V.P. Canarias: 355.—Ptas.

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	14
■ TRUCOS Y RUTINAS BASICAS	21
■ EL TALLER DEL HARDWARE	32
■ APRENDER CON EL ORDENADOR	39
■ PEQUEÑA HISTORIA DE LA INFORMATICA	50
■ TEMAS MONOGRAFICOS DE VANGUARDIA	53
■ TERMINOLOGIA	57
■ VOCABULARIO DE INFORMATICA	59

EXPERIENCIA Y PRACTICAS EN LOGO

■ Operaciones con objetos



ENOSULTIMO objeto
en forma abreviada:
MU objeto

Devuelve el objeto
especificado sin su último
elemento.

Si objeto es una pala-
bra, la devuelve sin el últi-

mo carácter.

```
? ESCRIBE MU "CAMION  
CAMIO
```

```
? ESCRIBE MU "A
```

Si a la letra A le quitamos el último ca-
rácter, no queda nada.

```
? HAZ "C MU "SABADO
```

```
? ESCRIBE :A
```

```
SABAD
```

```
? ES MU "
```

```
MENOSULTIMO NO ACEPTA COMO
```

DATO

La primitiva MENOSULTIMO no acepta
la palabra vacía como dato.

Si objeto es una lista, la primitiva ME-
NOSULTIMO la devuelve sin la última palabra
o lista.

```
? ES MU [ LOS LIBROS INTERESAN-  
TES ]  
LOS LIBROS
```

```
? ES MU [ REFRESCOS [ DE NARAN-  
JA ] ]  
REFRESCOS
```

```
? ES MU [A]
```

Al suprimir el último elemento de la lis-
ta especificada no queda nada que escribir.

```
? ES MU [ ]
```

```
MENOSULTIMO NO ACEPTA [ ] COMO
```

DATO

Veamos algunos ejemplos con esta pri-
mitiva.

```
? PARA P1
```

```
> ES ULTIMO MENOSULTIMO [ ESTA-  
MOS EN UNA ISLA DESIERTA ]
```

```
> FIN
```

```
? P1
```

```
ISLA
```

El último elemento de la lista especifi-
cada sin su último elemento es la palabra ISLA.

Hemos incluido en una sola orden dos
primitivas, si las separamos:

```
? PARA P2
```

```
> HAZ "A [ ESTAMOS EN UNA ISLA  
DESIERTA ]
```

```
> HAZ "B MENOSULTIMO :A
```

```
> ESCRIBE ULTIMO :B
```

```
> FIN
```

```
? P2
```

```
ISLA
```

obtenemos lo mismo que antes.

Si queremos suprimir los dos últimos
elementos de un objeto determinado:

```
? PARA E1
```

```
> HAZ "A [ EL SOL PUEDE QUEMAR  
LA PIEL ]
```

```
> ES MU MU : A
```

```
> FIN
```

***A cualquier lista podemos añadirle un objeto
Logo.***

EXPERIENCIA Y PRACTICAS EN LOGO

```
? El
EL SOL PUEDE QUEMAR
```

Obtenemos una lista que carece de los dos últimos elementos de la lista especificada.

— PONULTIMO objeto lista

en forma abreviada:

```
PU objeto lista
```

Devuelve una nueva lista formada por la que se ha especificado más el objeto colocado en la última posición.

```
? ES PU "LAPIZ "LARGO
PONULTIMO NO ACEPTA LARGO
COMO DATO
```

El segundo dato que acompaña a la primitiva PU debe ser obligatoriamente una lista.

```
? ES PU "NIÑOS [ LOS MAS ]
LOS MAS NIÑOS
```

El objeto en este ejemplo es la palabra NIÑOS y la lista [LOS MAS], la lista obtenida es la especificada más el objeto colocado en la última posición:

```
LOS MAS NIÑOS
```

```
? ES PU [ EL ORDENADOR ] [ HA DESAPARECIDO ]
```

```
HA DESAPARECIDO [ EL ORDENADOR ]
```

```
? ES PU [ ] [ LA VENTANA ]
LA VENTANA [ ]
```

Veamos algunos ejemplos con esta primitiva.

```
? PARA A1
> HAZ "A UL PU [ LA PELICULA ] [ ES DE TERROR ]
> ES :A
> FIN
```

El procedimiento A1 devuelve la lista [LA PELICULA], que es el último elemento de la lista obtenida al colocar el objeto [LA PELICULA] detrás de la lista [ES DE TERROR].

Compruébalo tecleando:

```
? A1
LA PELICULA
```

El siguiente ejemplo utiliza dos veces seguidas la primitiva PU.

```
? PARA A2
> ES PU PU "CASI [ ES DE NOCHE ]
[ ESTA OSCURECIENDO ]
> FIN
```

Si lo ejecutamos:

```
? A2
```

obtenemos la lista:

```
ESTA OSCURECIENDO [ ES DE NOCHE CASI ]
```

Si para clasificando queremos separarlas:

```
? PARA A3
> HAZ "A PU "CASI [ ES DE NOCHE ]
> HAZ "B PU :A [ ESTA OSCURECIENDO ]
> ES :B
> FIN
```

```
? A3
ESTA OSCURECIENDO [ ES DE NOCHE CASI ]
```

NOTA: La sintaxis de esta primitiva en el LOGO del SPECTRUM es:

```
PONULT objeto [ lista ]
```

■ Variables para la primitiva punto

Cuando estudiamos la primitiva punto, vimos que la sintaxis que tenía era:

```
? PUNTO [ x y ]
```

Los valores de x e y debían ser dos números fijos correspondientes a alguno de los puntos en que se divide la pantalla.

Por ejemplo, el punto (0,0) corresponde al punto central de la pantalla.

Los valores de la primitiva PUNTO no pueden ser en principio valores que tengan asignados dos variables determinadas.

Si se introducen las órdenes siguientes:

```
? HAZ "X 50
? HAZ "Y 50
? PUNTO [ :X :Y ]
```

obtendríamos el mensaje:

```
PUNTO NO ADMITE :X COMO DATO
```

Podemos suprimir el último elemento de cualquier objeto Logo.

Sin embargo, el Logo nos permite el utilizar variables con esta primitiva.

Para ello lo que tenemos que hacer es convertir los valores en una lista. Esto lo logramos introduciendo la primitiva LISTA:

? PUNTO LISTA :X :Y

Se indica a continuación de PUNTO LISTA seguida de las dos variables. En este caso no hay que escribir los corchetes.

Por ejemplo:

? PUNTO LISTA [:X :Y]

nos devolverá el mensaje:

PUNTO NO ACEPTA [[:X :Y]] COMO DATO

De esta forma podemos conseguir que el valor de los datos de PUNTO sea, por ejemplo, el resultado de una serie de operaciones realizadas con valores que estén asignados a variables.

También es posible utilizar una variable para una de las coordenadas y la otra indicarla directamente:

? HAZ "X 100

? PUNTO LISTA :X 50

En este caso tampoco es necesario indicar los corchetes.

El siguiente ejemplo dibuja, aleatoriamente, puntos en la pantalla:

? HAZ "X AZAR 200

? HAZ "Y AZAR 150

? PUNTO LISTA :X :Y

■ Cuadro resumen

— MONOSULTIMO objeto

o

MU objeto

Devuelve el objeto especificado sin su último elemento.

— PONULTIMO objeto lista

o

PU objeto lista

Devuelve una nueva lista formada por la lista especificada y el objeto como último elemento.

NOTA. La sintaxis en el SPECTRUM es:

PONULT objeto [lista]

■ Ejercicios

1. Dibuja con la tortuga un reloj de arena.
2. Realiza el siguiente dibujo:

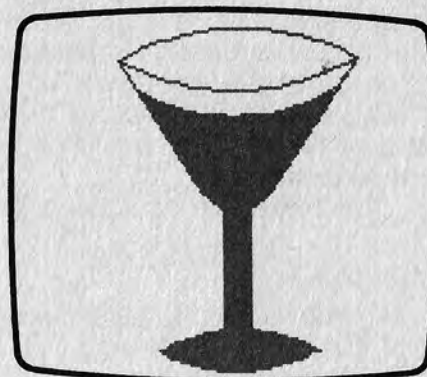


Fig. 1.

3. Realiza el siguiente dibujo:

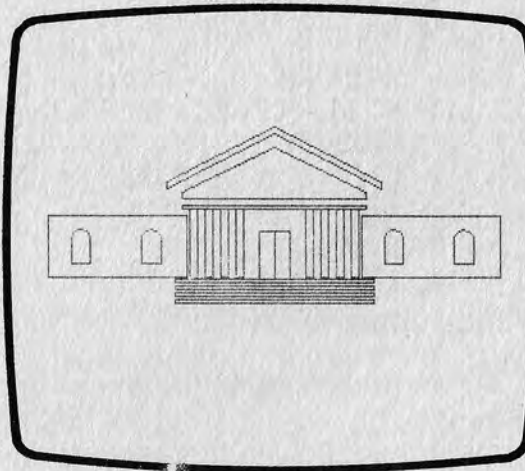


Fig. 2.

4. Define los procedimientos necesarios para realizar el siguiente proceso:

1) Solicitar que se introduzcan los coeficientes de las incógnitas de una ecuación de

El segundo dato que acompaña a la primitiva PONULTIMO debe ser obligatoriamente una lista.

EXPERIENCIA Y PRACTICAS EN LOGO

la forma:

$$aX + bY + c = 0$$

2) Una vez introducidos y tras pulsar una tecla, aparecerá en la pantalla la ecuación introducida, unos ejes de coordenadas y una tabla para introducir valores.

3) Solicitar que se introduzcan 17 valores para la incógnita X desde la tabla. Estos valores no pueden ser mayores a 15 ni menores a -15. Una vez introducido un valor, aparecerá el correspondiente valor de Y también en la tabla, a la vez que se representa el punto correspondiente en los ejes de coordenadas.

4) Una vez introducidos los 17 valores se dará la posibilidad de poder introducir otra ecuación si se desea.

5. ¿Son correctas las siguientes órdenes?:

- ? PARA A1
 - > HAZ "A FRASE "LAS "CASAS
 - > HAZ "B PU "SON :A
 - > ES :B
 - > FIN
- ? PARA A2
 - > HAZ "A PALABRA "CARA "COL
 - > HAZ "B PU [LENTO] :A
 - > ES :B
 - > FIN
- ? PARA A4
 - > ES MU (FRASE [ES ASI O NO] "ES "ASI)
 - > FIN
 - ? HAZ "A 10
 - ? HAZ "B 20
 - ? PUNTO [:A :B]
- ? HAZ "A 15
 - ? "B 20
 - ? PUNTO LISTA (:A*10) (:B - :A)

■ Solución a los ejercicios

1:

Realizamos el dibujo del reloj de arena con el siguiente procedimiento:

- ? PARA RELOJ
- > BP

- > SL
- > PONPOS [-30 60]
- > BL
- > PONCL 2
- > PONRUMBO 120
- > REPITE 32 [AV 2 GI 2]
- > PONRUMBO 300
- > REPITE 32 [AV 2 GI 2]
- > PONRUMBO 150
- > AV 60
- > RE 30
- > HAZ "A COORX
- > HAZ "B COORY
- > AV 30
- > PONRUMBO 180
- > PONCL 1
- > AV 30
- > RE 30
- > PONRUMBO 30
- > PONCL 2
- > AV 60
- > SL
- > PONPOS [-30 -40]
- > BL
- > PONRUMBO 120
- > REPITE 32 [AV 2 GI 2]
- > PONRUMBO 330
- > AV 60
- > PONRUMBO 210
- > AV 60
- > RE 10
- > PONRUMBO 60
- > PONCL 1
- > REPITE 32 [AV 1.7 GD 21]
- > SL
- > PONX COORX - 5
- > BL
- > RELLENA
- > SL
- > PONX :A PONY :B
- > PONRUMBO 60
- > BL
- > REPITE 32 [AV 1 GD 2]
- > SL
- > PONX COORX - 3
- > BL
- > RELLENA
- > FIN

Para ejecutar el procedimiento teclea:

- ? RELOJ

La primitiva MENOSULTIMO no acepta la palabra ni la lista vacía como datos.

te aparecerá un dibujo como éste:

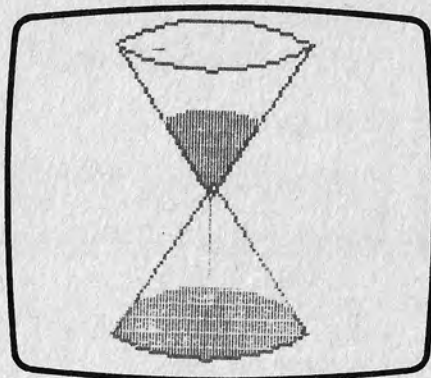


Fig. 3.

2:

Dibujamos la copa con dos procedimientos COPA y CIR, este último es llamado desde COPA.

```
? PARA COPA ,
> BP
> SL
> PONPOS [ -40 60 ]
> BL
> PONCL 3
> PONRUMBO 120
> REPITE 32 [ AV 3 GI 2 ]
> PONRUMBO 300
> REPITE 32 [ AV 3 GI 2 ]
> PONRUMBO 150
> AV 65
> RE 50
> PONRUMBO 120
> PONCL 2
> REPITE 32 [ AV 2.5 GI 2 ]
> PONCL 3
> PONRUMBO 210
> RE 15
> AV 65
> CIR 0.9
> SL
> PONRUMBO 0
> AV 20
> PONCL 2
> BL
> RELLENA
> SL
```

```
> RE 20
> PONRUMBO 180
> BL PONCL 3
> REPITE 2 [ AV 50 GD 90 AV 10 GD
  90 ]
> SL
> PONRUMBO 190
> AV 5
> BL
> RELLENA
> SL
> PONPOS [ -25 -63 ]
> BL
> PONRUMBO 120
> REPITE 32 [ AV 2 GI 2 ]
> PONRUMBO 300
> REPITE 32 [ AV 2 GI 2 ]
> SL
> PONX COORX + 4
> BL
> RELLENA
> FIN
```

El procedimiento CIR se repite hasta que el valor de la variable A es 0.3.

```
? PARA CIR :A
> PONRUMBO 240
> REPITE 30 [ AV :A GD 2 ]
> PONRUMBO 180
> AV 1
> PONRUMBO 120
> HAZ "A :A - 0.1
> REPITE 30 [ AV :A GI 2 ]
> HAZ "A :A - 0.1
> PONRUMBO 180 AV 1
> SI :A = 0.3 [ ALTO ] [ CIR :A ]
> FIN
```

Para ver el dibujo de la copa teclea:

```
> ? COPA
```

3:

Para realizar el dibujo definimos varios procedimientos que van a dibujar diferentes partes de él.

En primer lugar, definimos dos procedimientos para dibujar las columnas verticales.

```
? PARA COLUM
> SL
```

Para suprimir dos elementos de un objeto Logo podemos poner dos primitivas (MENOSULTIMO) seguidas.

EXPERIENCIA Y PRACTICAS EN LOGO

```
> PONPOS [ -55 -10 ]
> REPITE 4 [ A ]
> SL
> PONPOS [ 20 -10 ]
> REPITE 4 [ A ]
> FIN
```

Este procedimiento sitúa a la tortuga en los lugares correspondientes para dibujar las columnas y a continuación realiza cuatro llamadas al procedimiento A, que es el que las dibuja:

```
? PARA A
> BL
> PONRUMBO 0
> REPITE 2 [ AV 50 GD 90 AV 5 GD 90 ]
> SL
> GD 90 AV 10
> FIN
```

Las escaleras las dibuja el siguiente procedimiento:

```
? PARA ESCA
> SL
> PONPOS [ -65 -11 ]
> PONRUMBO 90
> REPITE 4 [ REPITE 2 [ BL
  AV 130 GD 90 AV 3 GD 90 ]
  SL GD 90 AV 5 GI 90 ]
> FIN
```

Para dibujar la puerta definimos PUER-

TA

```
? PARA PUERTA
> SL
> PONPOS [ -10 -11 ]
> PONRUMBO 0
> REPITE 2 [ AV 35 GD 90 AV 20 GD 90 ]
> GD 90 AV 10 GI 90 AV 35
> FIN
```

La parte superior del monumento la dibuja:

```
? PARA SUPE
> SL
> PONPOS [ -60 40 ]
> PONRUMBO 90
> REPITE 2 [ AV 120 GI 90 AV 2 GI 90 ]
> PONPOS [ -60 48 ]
```

```
> PONRUMBO 0
> BL
> AV 5
> PONPOS [ 0 85 ]
> PONPOS [ 60 53 ]
> RE 5
> GI 90 AV 120
> SL
> PONPOS [ -70 55 ]
> BL
> PONPOS [ 0 95 ]
> PONPOS [ 70 55 ]
> PONRUMBO 0
> AV 5
> PONPOS [ 0 100 ]
> PONPOS [ -70 60 ]
> RE 5
> FIN
```

Por último, para dibujar las partes laterales definimos el siguiente procedimiento:

```
? PARA LATE
> PONRUMBO 0
> BL
> REPITE 2 [ AV 45 GD 90
  AV 90 GD 90 ]
> SL
> AV 10 GD 90 AV 15
> GI 90
> REPITE 2 [ BL AV 20
  REPITE 5 [ GD 36 AV 4 ]
  AV 16 GD 90 AV 12 SL
  RE 45 GD 90 ]
> FIN
```

Una vez que tenemos todos los procedimientos definidos, los englobamos en uno para inicializar el proceso y realizar el dibujo completo:

```
? PARA MONUMENTO
> BP
> OT
> COLUM
> PUERTA
> ESCA
> SUPE
> SL
> PONPOS [ -147 -10 ]
> LATE
> SL
> PONPOS [ 57 -10 ]
```

La primitiva PUNTO puede utilizar variables como datos.

```
> LATE
> FIN
```

Para ejecutarlo tenemos que introducir:

```
? MONUMENTO
```

4:

En primer lugar, definimos el procedimiento que nos permite introducir el coeficiente de la X, de la Y y del término independiente:

```
? PARA INT
> BP
> BT
> PONCURSOR [ 2 2 ]
> ES [ INTRODUCIR EL COEFICIENTE
DE LA X ]
> ES "
> HAZ "X LL
> SI :X = [ ] [ INT ]
> HAZ "X UL :X
> PONCURSOR [ 2 6 ]
> ES [ INTRODUCIR EL COEFICIENTE
DE LA Y ]
> ES "
> HAZ "Y LL
> SI :Y = [ ] [ INT ]
> HAZ "Y UL :Y
> PONCURSOR [ 2 10 ]
> ES [ INTRODUCIR EL TERMINO IN-
DEPENDIENTE ]
> ES "
> HAZ "T LL
> SI :T = [ ] [ INT ]
> HAZ "T UL :T
> FIN
```

En la pantalla van apareciendo los mensajes de petición de valores.

A diferentes variables (X, Y, T) vamos asignándoles, respectivamente, los diferentes valores introducidos.

Al utilizar la primitiva LL para pedir los valores, las variables contienen los datos en forma de lista, por lo que tenemos que transformarlas para poder operar, ya que no se pueden realizar operaciones aritméticas con variables con contengan listas.

Esto lo conseguimos asignándole a la misma variable el elemento de la lista. (Ya que

sólo contiene uno, podemos, por ejemplo, asignarle el último.)

Se pregunta para cada dato si lo que se ha introducido es igual a la lista vacía ([]), lo que ocurre cuando se pulsa RETURN sin introducir ningún valor. En el caso de que así sea, se vuelve a solicitar los datos.

Una vez que se ejecute este procedimiento, en la parte inferior aparece el mensaje PULSA UNA TECLA, permaneciendo el proceso parado hasta que una tecla no sea pulsada.

Esto lo conseguimos con el procedimiento T:

```
? PARA T
> PONCURSOR [ 12 22 ]
> ES [ PULSA UNA TECLA ]
> HAZ "TECLA LC
> BP
> BT
> FIN
```

Una vez que se haya pulsado una tecla, la pantalla se borrará y aparecerá entonces en la parte superior de la pantalla la ecuación que se ha introducido, a la derecha los ejes de coordenadas y a la izquierda de los ejes, una tabla en donde se van a ir introduciendo los valores para la X.

El procedimiento que escribe la ecuación completa con los datos introducidos es:

```
? PARA A
> PONCURSOR [ 2 0 ]
> ES [ LA ECUACION INTRODUCIDA
ES ]
> PONCURSOR [ 2 2 ]
> ES (FRASE :X "X "+ :Y "Y "+ :T "=" )
> FIN
```

Para dibujar los ejes de coordenadas definimos el procedimiento:

```
? PARA EJES
> SL
> PONX 40
> BL
> V 60 RE 120
> AV 5 GD 90
> RE 1
> REPITE 23 [ BL AV 2 RE 2 SL GI 90
AV 5 GD 90 ]
```

No es necesario que los dos datos de PUNTO sean variables.

EXPERIENCIA Y PRACTICAS EN LOGO

```
> SL
> PONPOS [ 40 0 ]
> PONRUMBO 90
> BL
> AV 80 RE 160
> AV 5 GI 90 RE 1
> REPITE 31 [ BL AV 2 RE 2 SL GD 90
  AV 5 GI 90 ]
> PONCURSOR [ 25 5 ]
> ES "Y
> PONCURSOR [ 35 13 ]
> ES "X
> FIN
```

Los ejes aparecen divididos en intervalos de cinco puntos y el centro de coordenadas lo situamos en el punto de la pantalla (40, 0).

La tabla la dibuja el procedimiento TABLA:

```
? PARA TABLA
> SL
> PONPOS [ -130 -60 ]
> PONRUMBO 0
> BL
> REPITE 2 [ AV 120 GD 90 AV 70 GD
  90 ]
> AV 140 GD 90
> AV 70 GD 90
> AV 20 GD 90
> AV 35 GD 90
> AV 20 RE 140
> PONCURSOR [ 5 5 ]
> ES "X
> PONCURSOR [ 10 5 ]
> ES "Y
> FIN
```

DATOS escribe en la pantalla el mensaje de petición de valores para la incógnita X:

```
? PARA DATOS
> PONCURSOR [ 2 22 ]
> ES [ DA VALORES A LA INCOGNITA X ]
> HAZ "F 7
> FIN
```

Todo está preparado para comenzar a dar valores a la ecuación y que se vayan representando los puntos correspondientes.

El procedimiento F solicita 17 valores

para la incógnita X. Una vez que se ha introducido un valor y controlado que no es la lista vacía, ni que es superior a 15 ni menor que -15, llamamos al procedimiento CAL, que es el que se encarga de calcular el valor correspondiente a Y y de dibujar el punto en los ejes de coordenadas.

Una vez realizada esta operación se pregunta si el valor que se va a introducir es el decimotercero. Si no lo es, se vuelve a ejecutar el procedimiento, y si lo es, se ejecuta el procedimiento OTRO, que pregunta si se desea introducir otra ecuación.

```
? PARA F
> PONCURSOR LISTA 4 :F
> HAZ "X1 LL
> SI :X1 - [ ] [ F ]
> HAZ "X1 UL :X1
> SI O :X1 > 15 :X1 < -15 [ F ]
> CAL
> HAZ "F :F + 1
> SI :F = 18 [ OTRO ] [ F ]
> FIN
```

El procedimiento CAL calcula, en primer lugar, el valor de Y, y a continuación lo escribe, redondeando, en el lugar correspondiente en la tabla.

A continuación dibuja el punto correspondiente:

```
? PARA CAL
> HAZ "RY (-:T - (:X*X1)) / :Y
> PONCURSOR LISTA 9 :F
> ES REDONDEA :RY
> PUNTO LISTA ((:X1* 5) + 40) (:R* 5)
> FIN
```

Para calcular el punto, multiplicamos el valor de la X por cinco (intervalo de la división de los ejes) y le sumamos 40, que es el desplazamiento del centro de los ejes. La Y sólo la multiplicamos por 5, ya que desplazamiento con respecto al eje Y no hay:

OTRO nos permite la posibilidad de introducir otra ecuación:

```
? PARA OTRO
> BT
> PONCURSOR [ 0 22 ]
> ES [ ¿INTRODUCES OTRA ECUACION? S/N ]
```

Si utilizas variable en PUNTO, no te olvides que tienes que escribir delante de las variables LISTA.

```

> HAZ "SN LC
> SI :SN = "S [ ECU ]
> SI :SN = "N [ BP BT ALTO]
> OTRO
> FIN

```

Agrupamos en ECU todos los procedimientos:

```

? PARA ECU
> BP
> BT
> OT
> INT
> T
> A
> EJES
> TABLA
> DATOS
> F
> FIN

```

Para ejecutarlo introducimos:

```

? ECU

```

5:

```

? PARA A1
> HAZ "A FRASE "LAS "CASAS
> HAZ "B PU "SON :A
> ES :B
> FIN

```

CORRECTO

En la pantalla aparece la lista:

LAS CASAS SON.

```

— ? PARA A2
> HAZ "A PALABRA "CARA "COL
> HAZ "B PU [ LENTO ] :A
> ES :B
> FIN

```

INCORRECTO

El segundo dato de la primitiva PONULTIMO es una palabra (CARACOL) y esta primitiva sólo acepta, de segundo dato, listas.

El LOGO devuelve el mensaje de error:

PONULTIMO NO ACEPTA CARACOL COMO DATO

Si sería correcto:

```

? PARA A3
> HAZ "A PALABRA "CARA "COL
> HAZ "B PU :A [ LENTO ]
> ES :B
> FIN

```

```

— ? PARA A4
> ES MU (FRASE [ ES ASI O NO ] "ES "ASI)
> FIN

```

CORRECTO

En la pantalla aparece:

```

ES ASI O NO ES
? HAZ "A 10
? HAZ "B 20
? PUNTO [ :A :B ]

```

INCORRECTO

La primitiva PUNTO no admite variables si no se utiliza la orden LISTA.

```

? HAZ "A 15
? HAZ "B 20
? PUNTO LISTA (:A x 10) (:b - :A)

```

CORRECTO

Se dibujará un punto en la posición que se obtenga de las operaciones que realizan.

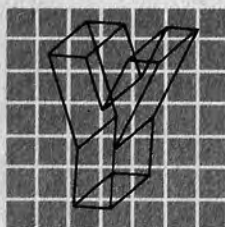
Puedes verificar al mismo tiempo varias primitivas que operen con los objetos.

Puedes indicar como dato para PUNTO una expresión aritmética.

Si utilizas LISTA en la primitiva PUNTO no puedes escribir los corchetes.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

Alta resolución



A hemos visto todo lo que se puede ver sobre la baja resolución. A partir de este momento vamos a meternos en el mundo de la alta resolución, en el mundo del dibujo.

Si en baja resolución podíamos realizar movimientos, en alta resolución también los podremos realizar, pero mucho más suaves.

Si en baja resolución podíamos dibujar decorados para nuestros programas, en alta resolución dichos decorados podrán tener una calidad muy superior.

Todo lo que hemos hecho con la baja resolución lo podemos realizar en alta resolución obteniendo unos resultados mucho más llamativos. La única dificultad es que trabajar en alta resolución es más difícil. Por ello, y debido a que cada ordenador tiene su propia manera de tratarla, vamos a separar en tomos distintos cada uno de los ordenadores. Así la explicación quedará más clara y detallada.

Alta resolución en el IBM pc y compatibles

Ante todo hay que decir que se supone que trabajaremos bajo GWBASIC y que la tarjeta gráfica de que disponemos es la normal. No vamos a hablar de ninguna otra, ya que hay muchas en el mercado y, normalmente, todo el mundo tiene la que suministra IBM con sus ordenadores.

La pantalla de alta resolución

Hemos visto en multitud de ocasiones que el IBM tiene tres tipos distintos de pantalla. De ellos, los dos últimos son de alta resolución. Por supuesto, en todos se puede escribir texto, pero se denomina «pantalla de texto» a la que resulta de ejecutar la orden SCREEN 0. Según esto, si escribimos SCREEN 1 o SCREEN 2, nos encontramos en alta resolución.

Pero ambos tipos de pantalla no son iguales. El primero de ellos tiene menos resolución que el segundo, pero permite la utilización de más colores. Así pues, si realizamos una tabla comparativa para ver la resolución de cada tipo de pantalla y sus colores nos quedaría lo siguiente:

PANTALLA	RESOLUCION VERTICAL	RESOLUCION HORIZONTAL	C. PAPEL	C. TINTA
SCREEN 1	200 PIXELS	320 PIXELS	TODOS	TRES
SCREEN 2	200 PIXELS	640 PIXELS	UNO	UNO

Fig. 1. Tabla comparativa de los dos modos de pantalla de alta resolución de que disponemos en el IBM.

En alta resolución no hablamos de caracteres, sino de pixels. Un pixel es un punto en la pantalla que puede estar apagado (0) o encendido (1). Si queremos realizar una correspondencia entre el número de caracteres de un tipo de pantalla y el número de pixels que tiene, podemos apreciar que cada carácter debe de estar compuesto por 8 pixels. Para llegar a esta conclusión no hace falta nada más que dividir el número de pixels en horizontal por el número de caracteres por línea de un tipo de pantalla. Si tenemos que en SCREEN 1 tenemos 320 pixels en horizontal y 40 caracteres por línea, al dividir 320 entre 40 nos da 8. Lo mismo ocurre si dividimos 640 por 80 (en el caso de SCREEN 2).

Estos pixels, como todo en el ordenador, están numerados. Es normal que sea así para poder referirnos a ellos. Así, los pixels horizontales se numeran desde 0 hasta 319 (o 639 en el caso de SCREEN 2) y los verticales desde 0 hasta 199. El origen de coordenadas lo tenemos localizado en la esquina superior izquierda de la pantalla. De todas maneras, esto lo podemos variar a nuestro gusto. Ya veremos más adelante cómo.

La diferencia entre SCREEN 0 y SCREEN 1 o SCREEN 2 es que mientras en la primera no podemos iluminar o apagar un punto específico de la pantalla, en la segunda sí podemos hacerlo. Esto nos permitirá dibujar con muchísimo más detalle, pudiendo incluso hacer dibujos que parezcan fotografías.

■ Los comandos de alta resolución

Por supuesto, para poder iluminar puntos, y para apagarlos, necesitamos una serie de comandos nuevos que nos lo permitan. El primero que vamos a ver nos permite iluminar un punto de la pantalla en el color que nosotros deseemos. Sólo tenemos que dar las coordenadas de dicho punto.

PSET (X,Y),C

Iluminaremos el punto que tiene las coordenadas de pantalla X,Y. Lo iluminaremos con el color C. Si dicho color no se pone, el ordenador le pondrá el color asignado con la última sentencia COLOR.

PRESET (X,Y),C

Nos apaga el punto que tiene como coordenada X e Y con el color C. Si el color C

no se especifica, el ordenador pondrá dicho punto del color del papel.

Aunque pueda parecer que estos dos comandos realizan la misma función, esto no es cierto. PSET, si no le damos el color, siempre nos dibujará un punto, mientras que PRESET, al no darle el color, nos lo borrará. Esto nos permite iluminar y apagar puntos sin tener en cuenta el color de fondo ni de tinta.

El programa 1 nos dibuja una serie de puntos con la instrucción PSET y luego nos los borra con PRESET.

```

1000 REM *****
1010 REM * NOCHE ESTRELLADA *
1020 REM *****
1030 REM
1040 REM *****
1050 REM * PROGRAMA DEMOSTRACION PARA VER EL *
1060 REM * USO DEL PSET Y PRESET EN EL IBM *
1070 REM *****
1080 REM
1090 SCREEN 1
1100 CLS
1110 OPTION BASE 1
1120 DIM XX(500)
1130 DIM YY(500)
1140 REM
1150 REM *** DIBUJO DEL CIELO ESTRELLADO ***
1160 REM
1170 LOCATE 25,1
1180 PRINT "DIBUJANDO EL CIELO":
1190 RANDOMIZE TIMER
1200 FOR I=1 TO 500
1210 LET X=INT(RND*520)
1220 LET Y=INT(RND*200)
1230 LET C=INT(RND*3+1)
1240 PSET(X,Y),C
1250 LET XX(I)=X
1260 LET YY(I)=Y
1270 NEXT I
1280 REM
1290 REM *** BUCLE DE RETARDO ***
1300 REM
1310 FOR I=1 TO 1500
1320 NEXT I
1330 REM
1340 REM *** BORRADO DEL CIELO ESTRELLADO ***
1350 REM
1360 BEEP:BEEP
1370 LOCATE 25,1
1380 PRINT "BORRANDO EL CIELO":
1390 FOR I=1 TO 500
1400 LET X=XX(I)
1410 LET Y=YY(I)
1420 PRESET(X,Y)
1430 NEXT I
1440 LOCATE 1,1:G

```

La figura 2 nos muestra la pantalla de nuestro IBM en una noche estrellada.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS



Fig. 2. Ejemplo de PSET y PRESET con el programa 1.

Pero no es éste el único comando que tenemos para dibujar en alta resolución. Si tuviésemos que dibujar líneas rectas, curvas y circunferencias punto a punto, nuestros programas serían muy lentos y de poco nos serviría la alta resolución. Para realizar estas funciones, y algunas más, disponemos de otra serie de comandos. Estos son:

LINE (X1,Y1)-(X2,Y2),C,T

Con este comando podemos trazar líneas rectas desde la coordenada X1,Y1 hasta la coordenada X2,Y2. Podemos incluir el color con que queremos que se dibuje la línea con solo poner una coma detrás del último paréntesis y el número de color (o con variable).

Este mundo es más versátil de lo que parece. Con él podemos dibujar rectángulos con sólo incluir el segundo parámetro (Tipo). Si, por ejemplo, escribimos:

LINE (0,0)-(100,100),1,B

El ordenador nos dibujará los contornos de un rectángulo cuya esquina superior izquierda es (0,0) y la esquina inferior derecha es (100,100) con el color 1.

Si lo que escribimos es:

LINE (0,0)-(100,100),2,BF

Lo que el ordenador dibujará será un rectángulo relleno de color, dos en la misma posición que el anterior.

Como ejemplo podemos introducir el programa 2. Con él podremos ver cómo se dibujan una serie de rectángulos de colores en nuestro ordenador.

```

1000 REM *****
1010 REM * NEW YORK *
1020 REM *****
1030 REM
1040 REM *****
1050 REM * PROGRAMA DEMOSTRACION PARA VER EL USO DEL *
1060 REM * COMANDO LINE EN EL IBM pc Y COMPATIBLES. *
1070 REM *****
1080 REM
1090 REM *** INICIALIZACION Y DIBUJO DE LA PANTALLA ***
1100 REM
1110 SCREEN 1
1120 CLS
1130 LOCATE 25,1
1140 PRINT "VISTA AEREA DE NEW YORK";
1150 LOCATE 1,1
1160 PRINT CHR$(201);STRING$(38,205);CHR$(187)
1170 FOR I=2 TO 15
1180   PRINT CHR$(186);TAB(40);CHR$(186)
1190 NEXT I
1200 PRINT CHR$(200);STRING$(38,205);CHR$(188)
1210 LET XMIN=15
1220 LET YMIN=15
1230 LET XMAX=800
1240 REM
1250 REM *** RECOGIDA DE OPCION ***
1260 REM
1270 RANDOMIZE TIMER
1280 LOCATE 20,1
1290 PRINT "COMO QUIERES LOS EDIFICIOS?"
1300 COLOR ,,,1:PRINT "L";:COLOR ,,,3:PRINT "LENOS O "
1310 COLOR ,,,1:PRINT "V";:COLOR ,,,3:PRINT "ACIOS"
  
```

```

1320 LET A$=INKEY$
1330 IF A$="" THEN GOTO 1320
1340 IF A$="L" OR A$="I" THEN LET SW=1:GOTO 1370
1350 IF A$="V" OR A$="v" THEN LET SW=0:GOTO 1370
1360 GOTO 1320
1370 LOCATE 20,1
1380 PRINT SPACE$(80)
1390 LOCATE 20,1
1400 PRINT "PULSA 'P' PARA PARAR"
1410 REM
1420 REM *** DIBUJO DE NEW YORK ***
1430 REM
1440 LET X1=INT(RND*XMAX+XMIN)
1450 LET Y1=INT(RND*100+YMIN)
1460 LET X2=INT(RND*XMAX+XMIN)
1470 LET Y2=120
1480 LET C=INT(RND*3+1)
1490 IF ABS(X2-X1)>80 THEN GOTO 1440
1500 IF SW=0 THEN LINE(X1,Y1)-(X2,Y2),C,B
1510 IF SW=1 THEN LINE(X1,Y1)-(X2,Y2),C,BF
1520 REM
1530 REM *** LECTURA DEL TECLADO Y RETARDO ***
1540 REM
1550 FOR I=1 TO 100
1560 IF INKEY$="P" THEN GOTO 1590
1570 NEXT I
1580 GOTO 1440
1590 CLS
1600 END

```

La figura 3 nos muestra un ejemplo de cómo se desarrollará el programa en nuestra pantalla.

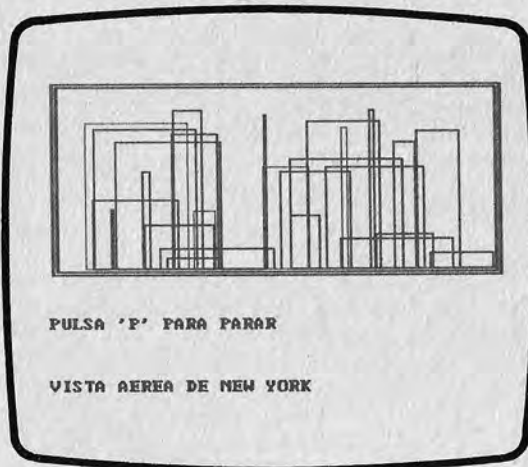


Fig. 3. El programa New York en plena acción.

CIRCLE (X,Y),R,C,A1,A2,S

Con este nuevo comando podremos dibujar círculos, elipses y arcos de una manera muy sencilla. En esta instrucción todos los parámetros son opcionales, a excepción de las

coordenadas del centro (X e Y) y del radio (R). Los parámetros, en el orden de aparición, son los siguientes:

X: Coordenada en X del centro de la circunferencia, elipse o arco a dibujar.

Y: Coordenada en Y del centro de la circunferencia, elipse o arco a dibujar.

R: Radio de la circunferencia, elipse o arco.

C: Color de la circunferencia, elipse o arco.

A1: Angulo (en radianes) del que partimos al dibujar un arco.

A2: Angulo (en radianes) al que llegamos al dibujar un arco.

S: Aspecto del círculo. Nos indica la relación entre el radio horizontal y el vertical. Nos permite dibujar elipses.

Se hace necesario explicar un poco más profundamente el uso del ángulo a la hora de dibujar arcos de circunferencia. El primer ángulo (A1) le dice al ordenador desde qué posición tiene que partir para dibujar un arco. El segundo ángulo le dice en qué ángulo tiene que parar. Al estar estos ángulos en radianes, y no en grados, se hace necesario el uso de una tabla. En la figura 4 aparece una tabla que

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

nos da la relación entre la cantidad de círculos a dibujar y su medición en radianes.

FRACCION DE UN CIRCULO	ANGULO APROXIMADO EN RADIANES
UN OCTAVO	0.785
UN CUARTO	1.47
UNA TERCERA PARTE	2.09
LA MITAD	3.14
DOS TERCERAS PARTES	4.19
TRES CUARTAS PARTES	4.71
ENTERO	6.28

Fig. 4. Relación entre las fracciones de un arco y sus grados medidos en radianes.

dios de un círculo son iguales. En el caso de que queramos dibujar una elipse, este valor no será 1, ya que la diagonal principal y la secundaria de un círculo siempre son distintas.

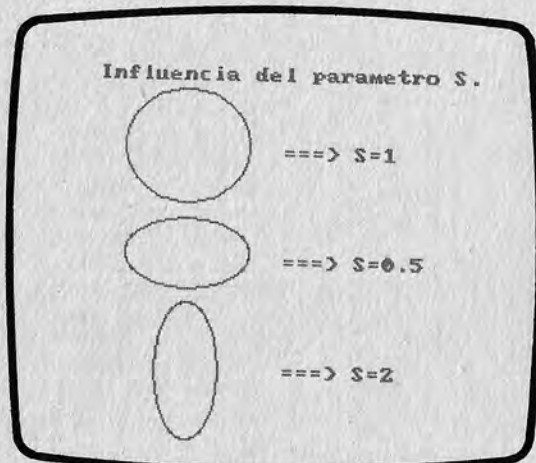


Fig. 5.

Con respecto al parámetro S, éste le dice al ordenador la relación entre el radio vertical y el horizontal del círculo a dibujar. Si lo que queremos dibujar es un círculo perfecto, dicha relación vale 1, ya que todos los ra-

Para entender mejor cómo hay que dibujar los círculos y las elipses en nuestro ordenador, vamos a ver un programa que nos dibujará espirales de varias formas utilizando círculos o elipses.

```

1000 REM *****
1010 REM * ESPIRALES *
1020 REM *****
1030 REM
1040 REM *****
1050 REM * PROGRAMA DE DEMOSTRACION PARA VER EL USO *
1060 REM * DEL COMANDO CIRCLE EN EL IBM pc. *
1070 REM *****
1080 REM
1090 REM *** INICIALIZACION ***
1100 REM
1110 LET XCEN=160
1120 LET YCEN=100
1130 SCREEN 1
1140 CLS
1150 KEY OFF
1160 REM
1170 REM *** ENTRADA DE OPCION ***
1180 REM
1190 LOCATE 10,1
1200 PRINT "PULSA UN NUMERO ENTRE 3 Y 9!"
1210 LET A$=INKEY$
1220 IF A$="" THEN GOTO 1210
1230 IF A$<"3" OR A$>"9" THEN GOTO 1210
1240 LET N=VAL(A$)
1250 LOCATE 13,1
1260 PRINT "QUE QUIERES?"
1270 PRINT
1280 PRINT "1 - CIRCULOS (S=1.0)"
1290 PRINT "2 - ELIPSES (S=0.5)"

```

```

1300 PRINT "3 - ELIPSES (S=2.0)"
1310 LET A$=INKEY$
1320 IF A$="" THEN GOTO 1310
1330 IF A$<"1" OR A$>"3" THEN GOTO 1310
1340 IF A$="1" THEN LET S=1
1350 IF A$="2" THEN LET S=.5
1360 IF A$="3" THEN LET S=2
1370 REM
1380 REM *** PROGRAMA PRINCIPAL ***
1390 REM
1400 CLS
1410 FOR I=0 TO 499 STEP .2
1420   LET X=XCEN+I*N*COS(I)
1430   LET Y=YCEN+I*N*SIN(I)
1440   CIRCLE(X,Y),10,...,S
1450 NEXT I
1460 FOR I=-1 TO 2000
1470 NEXT I
1480 RUN

```

El ejemplo de este programa en plena ejecución lo podemos ver en la figura 6.

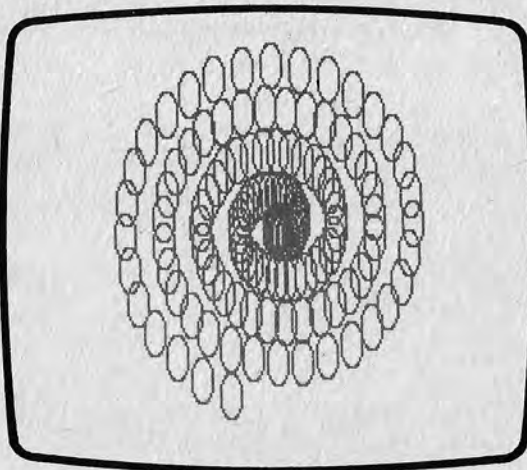


Fig. 6. Espirales.

PAINT (X,Y),C,B

Con este nuevo comando vamos a poder rellenar de color figuras cerradas, como pueden ser círculos, rectángulos y cuadrados, triángulos y, en general, cualquier forma cerrada, sin bordes abiertos, por complicada que ésta sea.

Todos los parámetros son opcionales, a excepción de las coordenadas. Su significado es el siguiente:

X: Coordenada en X desde donde queremos pintar.

Y: Coordenada en Y desde donde queremos pintar.

Ambas coordenadas han de estar situadas dentro de la figura que queremos rellenar de color.

C: Color del que queremos rellenar la figura.

B: Color del borde de la figura. El borde de la figura ha de ser de un mismo color, que puede ser igual o distinto a C. Si parte del contorno de la figura tiene un color distinto, entonces el dibujo se escapará de dentro de la figura.

A continuación, y ya para terminar, vamos a ver un programa que nos dibuja una serie de figuras y las rellena de color

```

1000 REM *****
1010 REM * RELLENOS *
1020 REM *****
1030 REM
1040 REM *****
1050 REM * PROGRAMA DEMO PARA VER EL USO *
1060 REM * DEL COMANDO PAINT EN EL IBM. *
1070 REM *****
1080 REM
1090 REM
1100 REM *** INICIALIZACION ***
1110 REM
1120 CLS
1130 SCREEN 1
1140 CLS
1150 KEY OFF
1160 REM
1170 REM *** COLOREADO DE UN CIRCULO ***
1180 REM
1190 FOR I=100 TO 104 STEP .5
1200   CIRCLE (160,100),I
1210 NEXT I
1220 GOSUB 3000 : REM PARA COLOREAR
1230 REM

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```
1240 REM *** COLOREADO DE UNA ELIPSE ***
1250 REM
1260 CLS
1270 FOR I=100 TO 104 STEP .5
1280   CIRCLE (160,100),I,,,2
1290 NEXT I
1300 GOSUB 3000 : REM PARA COLOREAR
1310 REM
1320 REM *** COLOREADO DE UN CUADRADO ***
1330 REM
1340 CLS
1350 FOR I=0 TO 4
1360   LINE (90-I,30-I)-(230+I,170+I),3,B
1370 NEXT I
1380 GOSUB 3000 : REM PARA COLOREAR
1390 END
3000 REM
3010 REM *****
3020 REM * SUBROUTINA DE COLOREADO *
3030 REM *****
3040 REM
3050 PAINT (160,100),1,3
3060 PAINT (160,100),2,3
3070 PAINT (160,100),3,3
3080 PAINT (160,100),0,0
3090 RETURN
```

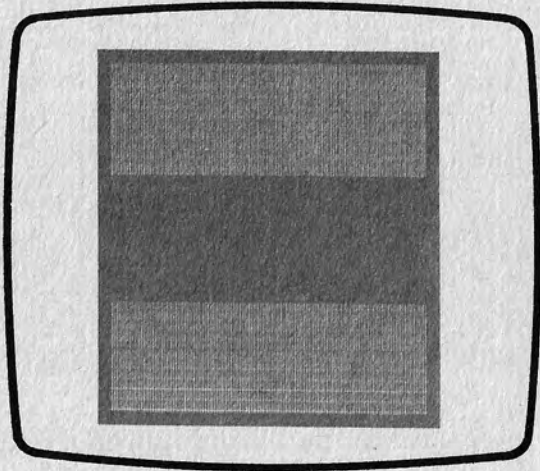
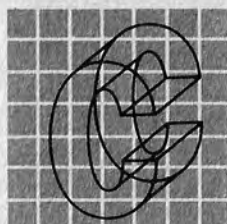


Fig. 7. Usando la instrucción PAINT.

TRUCOS Y RUTINAS BASICAS



OMO ya vimos en el tomo anterior, a la hora de trabajar con la memoria es necesario tener un programa que nos permite realizar una serie de funciones para poder trabajar con más comodidad. Todo lo que un programa nos puede permitir hacer también

podemos hacerlo nosotros mismos paso a paso, pero siempre resulta mucho más fácil y cómodo si utilizamos un programa que nos ayude.

Vimos en el tomo anterior un programa MONITOR de memoria que podíamos utilizar en todos los ordenadores compatibles IBM, así como en el AMSTRAD y en el COMMODORE. A continuación aparece dicho programa en versión para SPECTRUM.

```
1000 REM *****
1010 REM * MONITOR DE MEMORIA *
1020 REM *****
1030 REM
1040 REM *****
1050 REM *
1060 REM * (c) Ediciones Siglo Cultural *
1070 REM *
1080 REM * (c) 1987
1090 REM *
1100 REM *****
1110 REM
1120 REM *****
1130 REM * INICIALIZACION *
1140 REM *****
1150 REM
1160 GOSUB 3210
1170 LET BAS=10
1180 POKE 23286,0
1190 POKE 23858,8
1200 LET MP=0
1210 LET MOD=2
1220 REM
1230 REM *****
1240 REM * PROGRAMA PRINCIPAL *
1250 REM *****
1260 REM
1270 CLS
1280 PRINT AT 2,15;"ESTOY EN BASE ";BAS
1290 GOSUB 1480
1300 LET K$=INKEY$
1310 IF K$="" THEN GOTO 1300
1320 IF K$="6" THEN LET MP=MP+(1 AND MP<65535!)-(MP AND MP=65535!):GOSUB 1480:GO
TO 1300
```

TRUCOS Y RUTINAS BASICAS

```

1330 IF K$="7" THEN LET MP=MP-(1 AND MP>0)+(65535!-MP AND MP=0):GOSUB 1480:GOTO
1300
1340 IF K$="5" THEN LET MP=MP+(10 AND MP+10<65535!)-(10-65536! AND MP+10>65535!):
:GOSUB 1480:GOTO 1300
1350 IF K$="8" THEN LET MP=MP-(10 AND MP-10>0)+(-10+65526! AND MP-10<0):GOSUB 14
80:GOTO 1300
1360 IF K$="0" THEN INPUT "VALOR A POKEAR = ";P:POKE MP,P:GOTO 1270
1370 IF K$="H" THEN GOSUB 3520:GOTO 1270
1380 IF K$="B" THEN GOSUB 2010:GOTO 1300
1390 IF K$="N" AND PEEK(29900)<>0 THEN GOSUB 2180:GOTO 1300
1400 IF K$="J" THEN GOSUB 2250:GOTO 1270
1410 IF K$="L" THEN GOSUB 2350:GOTO 1270
1420 IF K$="S" THEN GOSUB 2560:GOTO 1270
1430 IF K$="M" THEN GOSUB 2680:GOTO 1270
1440 IF K$="X" THEN GOSUB 2770:GOTO 1270
1450 IF K$="P" THEN GOSUB 1720:GOTO 1270
1460 IF K$="T" THEN GOSUB 2820:GOTO 1270
1470 GOTO 1300
1480 REM
1490 REM *****
1500 REM * DISPLAY DE LA MEMORIA *
1510 REM *****
1520 REM
1530 PRINT AT 0,0;
1540 FOR A=MP-10 TO MP+11
1550   LET B=A
1560   IF A>65535! THEN LET A=A-65536!
1570   IF A<0 THEN LET A=65536!+A
1580   LET N$=STR$(A)
1590   LET P$=STR$(PEEK(A))
1600   IF MOD=1 THEN GOSUB 3100
1610   IF LEN(N$)<5 THEN LET N$="0"+N$:GOTO 1610
1620   IF MOD=2 THEN IF LEN(P$)<3 THEN LET P$="0"+P$:GOTO 1620
1630   IF MOD=3 THEN LET NUM=VAL(P$):GOSUB 1890
1640   PRINT " ";N$;" ";P$
1650   LET A=B
1660 NEXT A
1670 PRINT AT 10,0;">"
1680 IF MOD=1 THEN PRINT AT 10,15;"<"
1690 IF MOD=2 THEN PRINT AT 10,10;"<"
1700 IF MOD=3 THEN PRINT AT 10,9;"<"
1710 RETURN
1720 REM
1730 REM *****
1740 REM * SALIDA POR IMPRESORA *
1750 REM *****
1760 REM
1770 INPUT "DESDE LA DIR. ";I
1780 IF I<0 THEN GOTO 1770
1790 INPUT "HASTA LA DIR. ";F
1800 IF F<I OR F>65535! THEN GOTO 1790
1810 LET A=MP
1820 FOR M=I+10 TO F STEP 22
1830   LET MP=M
1840   GOSUB 1480
1850   COPY
1860 NEXT M
1870 LET MP=A
1880 RETURN
1890 REM
1900 REM *****
1910 REM * PASO DE DECIMAL A HEXADECIMAL *
1920 REM *****
1930 REM
1940 LET P$=""
1950 FOR C=1 TO 0 STEP -1

```

```

1960 LET N=INT(NUM/16^C)
1970 LET NUM=NUM-N*16^C
1980 LET P$=P$+(STR$(N) AND N<10)+("A" AND N=10)+("B" AND N=11)+("C" AND N=12
)+("D" AND N=13)+("E" AND N=14)+("F" AND N=15)
1990 NEXT C
2000 RETURN
2010 REM
2020 REM *****
2030 REM * BUSQUEDA DE BYTES *
2040 REM *****
2050 REM
2060 POKE 23728,0
2070 POKE 23729,0
2080 INPUT "CUANTOS BUSCO? (1-5) ";B
2090 IF B>5 OR B<1 THEN GOTO 2080
2100 POKE 29900,B
2110 FOR A=1 TO B
2120 CLS
2130 PRINT "DAME EL NUMERO ";A
2140 INPUT "NUMERO > ";C
2150 IF C>255 OR C<0 THEN GOTO 2140
2160 POKE 29900+A,C
2170 NEXT A
2180 LET A=USR 29500
2190 RANDOMIZE A+1
2200 POKE 23728,PEEK(23670)
2210 POKE 23729,PEEK(23671)
2220 LET MP=A
2230 GOSUB 1480
2240 RETURN
2250 REM
2260 REM *****
2270 REM * SALTO A UNA DIRECCION DE MEMORIA *
2280 REM *****
2290 REM
2300 INPUT "A QUE DIRECCION SALTO? (0-65535) ";J
2310 IF J<0 OR J>65535! THEN GOTO 2300
2320 INPUT "ESTAS SEGURO? (S/N) ";A$
2330 IF A$(1)="S" OR A$(1)="s" THEN PRINT USR J
2340 GOTO 3750
2350 REM
2360 REM *****
2370 REM * CARGAR BYTES (LOAD) *
2380 REM *****
2390 REM
2400 GOSUB 2450
2410 IF A$(1)="M" OR A$(1)="m" THEN LOAD "M";1;N$CODE D:RETURN
2420 IF A$(1)<>"C" AND A$(1)<>"c" THEN GOTO 2400
2430 LOAD D$CODE D
2440 RETURN
2450 REM
2460 REM *****
2470 REM * CASETE O MICRODRIVE? *
2480 REM *****
2490 REM
2500 INPUT FLASH 1;"C";FLASH 0;"ASETE O ";FLASH 1;"M";FLASH 0;"ICRO? ";LINE A$
2510 INPUT "NOMBRE? (0-10 CHRS.) ";N$
2520 IF LEN(N$)>10 THEN GOTO 2510
2530 INPUT "DIRECCION? ( >30000 ) ";D
2540 IF D<30000 OR D>65536! THEN GOTO 2530
2550 RETURN
2560 REM
2570 REM *****
2580 REM * GRABAR BYTES (SAVE) *
2590 REM *****
2600 REM

```

TRUCOS Y RUTINAS BASICAS

```

2610 GOSUB 2450
2620 INPUT "LONGITUD? ";L
2630 IF L>35535! OR L<0 THEN GOTO 2620
2640 IF A$(1)="M" OR A$(1)="m" THEN SAVE "M";1:N$CODE D,L:RETURN
2650 IF A$(1)<>"C" AND A$(1)<>"c" THEN GOTO 2610
2660 SAVE N$CODE D,L
2670 RETURN
2680 REM
2690 REM *****
2700 REM * CAMBIAR PUNTERO DE MEMORIA *
2710 REM *****
2720 REM
2730 INPUT "MEMORY = ";M
2740 IF M>65535! OR M<0 THEN GOTO 2730
2750 LET MP=M
2760 RETURN
2770 REM
2780 REM *****
2790 REM * CAMBIO DE BASE *
2800 REM *****
2810 REM
2820 PRINT AT 5,18;"1. BASE 2."
2830 PRINT AT 7,18;"2. BASE 10."
2840 PRINT AT 9,18;"3. BASE 16."
2850 LET K$=INKEY$
2860 IF K$="" THEN GOTO 2850
2870 IF K$<>"1" AND K$<>"2" AND K$<>"3" THEN GOTO 2850
2880 LET BAS<(2 AND K$="1")+(10 AND K$="2")+(16 AND K$="3")
2890 LET MOD=VAL(K$)
2900 LET K$=""
2910 RETURN
2920 REM
2930 REM *****
2940 REM * MUEVE BYTES *
2950 REM *****
2960 REM
2970 INPUT "DESDE? ";P
2980 IF P<0 OR P>65535! THEN GOTO 2970
2990 INPUT "HASTA? ";D
3000 IF D<0 OR D>65535! THEN GOTO 2990
3010 INPUT "LONGITUD? ";L
3020 IF L<0 OR L>65535! THEN GOTO 3010
3030 IF D<P+L AND D>P THEN POKE 29562,184:RANDOMIZE P+L:POKE 29900,PEEK(23670):P
OKE 29901,PEEK(23671):RANDOMIZE D+L:POKE 29902,PEEK(23670):POKE 29903,PEEK(23671)
)
3040 IF D<P+L AND D>P THEN RANDOMIZE L:POKE 29904,PEEK(*23670):POKE 29905,PEEK(2
3671):RANDOMIZE USR 29550:POKE 29562,176:RETURN
3050 RANDOMIZE P:POKE 29900,PEEK(23670):POKE 29901,PEEK(23671)
3060 RANDOMIZE D:POKE 29902,PEEK(23670):POKE 29903,PEEK(23671)
3070 RANDOMIZE L:POKE 29904,PEEK(23670):POKE 29905,PEEK(23671)
3080 RANDOMIZE USR 29550
3090 RETURN
3100 REM
3110 REM *****
3120 REM * PASO DE DECIMAL A BINARIO *
3130 REM *****
3140 REM
3150 POKE 29910,VAL(P$)
3160 LET P$=""
3170 FOR D=1 TO 8
3180 LET P$=P$+STR$(USR 29600)
3190 NEXT D
3200 RETURN
3210 REM
3220 REM *****
3230 REM * CODIGO MAQUINA *

```

```

3240 REM *****
3250 REM
3260 RESTORE 3470
3270 FOR A=0 TO 34
3280   READ B
3290   POKE 29500+A,B
3300 NEXT A
3310 RESTORE 3500
3320 FOR A=0 TO 13
3330   READ B
3340   POKE 29550+A,B
3350 NEXT A
3360 RESTORE 3510
3370 FOR A=0 TO 14
3380   READ B
3390   POKE 29600+A,B
3400 NEXT A
3410 RETURN
3420 REM
3430 REM *****
3440 REM * LINEAS DATA CON EL CODIGO MAQUINA *
3450 REM *****
3460 REM
3470 DATA 237,91,176,92,33,204,116,126,213,71,35,26,19,190,32,10
3480 DATA 16,248,209,237,93,176,92,66,75,201,209,19,237,75,176,92
3490 DATA 122,184,40,2,24,222,123,186,200,24,217
3500 DATA 42,204,116,237,91,206,116,237,75,208,116,237,176,201
3510 DATA 58,214,116,203,39,50,214,116,1,0,0,208,14,1,201
3520 REM
3530 REM *****
3540 REM * INSTRUCCIONES *
3550 REM *****
3560 REM
3570 CLS
3580 PRINT INVERSE 1;"** I N S T R U C C I O N E S **"
3590 PRINT
3600 PRINT
3610 PRINT FLASH 1;"B";FLASH 0;"USCA BYTES EN MEMORIA"
3620 PRINT FLASH 1;"H";FLASH 0;"ELP. PONE ESTAS INSTRUCCIONES."
3630 PRINT FLASH 1;"J";FLASH 0;"UMP. EJECUTA CODIGO MAQUINA."
3640 PRINT FLASH 1;"L";FLASH 0;"OAD. CARGA BYTES."
3650 PRINT FLASH 1;"M";FLASH 0;"EMORY. AJUSTA EL PUNTERO."
3660 PRINT FLASH 1;"P";FLASH 0;"RINTER. PASA POR IMPRESORA."
3670 PRINT FLASH 1;"S";FLASH 0;"AVE. GRABA BYTES."
3680 PRINT FLASH 1;"T";FLASH 0;"RASLADA BYTES."
3690 PRINT FLASH 1;"X";FLASH 0;" = CAMBIA DE BASE."
3700 PRINT FLASH 1;"5";FLASH 0;" = AUMENTA EL PUNTERO EN 10."
3710 PRINT FLASH 1;"6";FLASH 0;" = AUMENTA EL PUNTERO EN UNO."
3720 PRINT FLASH 1;"7";FLASH 0;" = DISMINUYE EL PUNTERO EN UNO."
3730 PRINT FLASH 1;"8";FLASH 0;" = DISMINUYE EL PUNTERO EN 10."
3740 PRINT FLASH 1;"O";FLASH 0;" = POKEA EN EL PUNTERO."
3750 PRINT #1;FLASH 1;"PULSA UNA TECLA"
3760 PAUSE 0
3770 CLS
3780 RETURN

```

La forma de funcionar no dista mucho de la de la versión de IBM. Por lo que no hace falta explicar su utilización.

Por fin, y ya para terminar, tenemos

este mismo programa en versión para COM-MODORE. Se utiliza de la misma manera que la versión del IBM.

TRUCOS Y RUTINAS BASICAS

```

1000 REM *****
1010 REM * MONITOR DE MEMORIA *
1020 REM *****
1030 REM
1040 REM *****
1050 REM *
1060 REM * (c) Ediciones Siglo Cultural *
1070 REM *
1080 REM * (c) 1987
1090 REM *
1100 REM *****
1110 REM
1120 REM *****
1130 REM * INICIALIZACION *
1140 REM *****
1150 REM
1160 OPEN 1,0,1
1170 OPEN 2,4,2
1180 LET CH=1
1190 LET BA=10
1200 LET MP=0
1210 LET MO=2
1220 LET SW=1
1230 REM
1240 REM *****
1250 REM * PROGRAMA PRINCIPAL *
1260 REM *****
1270 REM
1280 PRINT CHR$(147)
1290 POKE 214,1:POKE 211,22
1300 PRINT "ESTOY EN BASE ";BA
1310 GOSUB 1570
1320 GET K$
1330 IF K$="" THEN GOTO 1320
1340 IF K$="1" THEN LET MP=MP-1
1350 IF MP<0 THEN LET MP=55535
1360 IF K$="2" THEN LET MP=MP+1
1370 IF MP>55535 THEN LET MP=0
1380 IF K$="8" THEN LET MP=MP-10
1390 IF MP<0 THEN LET MP=55535+MP
1400 IF K$="9" THEN LET MP=MP+10
1410 IF MP>55535 THEN LET MP=MP-55535
1420 IF K$>"0" AND K$<"." THEN GOSUB 1570:GOTO 1320
1430 IF K$="0" THEN GOSUB 4490:INPUT "VALOR A POKEAR = ";P:IF P>255 OR P<0 THEN
GOTO 1430
1440 IF K$="O" THEN POKE MP,P:GOSUB 4490:GOTO 1280
1450 IF K$="A" THEN GOSUB 3890:GOTO 1280
1460 IF K$="B" THEN GOSUB 2190:GOTO 1320
1470 IF K$="N" THEN GOSUB 2380:GOTO 1320
1480 IF K$="J" THEN GOSUB 2550:GOTO 1280
1490 IF K$="L" THEN GOSUB 2710:GOTO 1280
1500 IF K$="S" THEN GOSUB 3040:GOTO 1280
1510 IF K$="M" THEN GOSUB 3210:GOTO 1280
1520 IF K$="X" THEN GOSUB 3320:GOTO 1280
1530 IF K$="E" THEN GOSUB 1820:GOTO 1280
1540 IF K$="T" THEN GOSUB 3470:GOTO 1280
1550 IF K$="D" THEN LET SW=-SW:PRINT CHR$(147):GOTO 1280
1560 GOTO 1320
1570 REM
1580 REM *****
1590 REM * DISPLAY DE LA MEMORIA *
1600 REM *****

```

```

1610 REM
1620 IF SW=-1 THEN GOTO 4200
1630 POKE 214,0:POKE 211,0
1640 FOR A=MP-10 TO MP+11
1650   LET B=A
1660   IF A>65535! THEN LET A=A-65536!
1670   IF A<0 THEN LET A=65536!+A
1680   LET N$=MID$(STR$(A),2)
1690   LET P$=MID$(STR$(PEEK(A)),2)
1700   IF MO=1 THEN LET NU=VAL(P$):GOSUB 3650
1710   IF LEN(N$)<5 THEN LET N$="0"+N$:GOTO 1710
1720   IF MO=2 THEN IF LEN(P$)<3 THEN LET P$="0"+P$:GOTO 1720
1730   IF MO=3 THEN LET NU=VAL(P$):GOSUB 2010
1740   PRINT#CH," ";N$;" ";P$
1750   LET A=B
1760 NEXT A
1770 POKE 214,10:POKE 211,0:PRINT ">"
1780 IF MO=1 THEN POKE 214,10:POKE 211,15:PRINT "<"
1790 IF MO=2 THEN POKE 214,10:POKE 211,10:PRINT "<"
1800 IF MO=3 THEN POKE 214,10:POKE 211,9:PRINT "<"
1810 RETURN
1820 REM
1830 REM *****
1840 REM * SALIDA POR IMPRESORA *
1850 REM *****
1860 REM
1870 GOSUB 4490:INPUT "DESDE LA DIR. ";I:GOSUB 4490
1880 IF I<0 THEN GOTO 1870
1890 GOSUB 4490:INPUT "HASTA LA DIR. ";F:GOSUB 4490
1900 IF F<I OR F>65535! THEN GOTO 1890
1910 LET A=MP
1920 FOR M=I TO F STEP 22
1930   LET MP=M
1940   LET CH=2
1950   GOSUB 1570
1960 NEXT M
1970 LET MP=A
1980 LET CH=1
1990 GOSUB 1570
2000 RETURN
2010 REM
2020 REM *****
2030 REM * PASO DE DECIMAL A HEXADECIMAL *
2040 REM *****
2050 REM
2060 LET P$=""
2070 FOR C=1 TO 0 STEP -1
2080   LET N=INT(NU/16^C)
2090   LET NU=NU-N*16^C
2100   IF N<10 THEN LET P$=P$+MID$(STR$(N),2,1)
2110   IF N=10 THEN LET P$=P$+"A"
2120   IF N=11 THEN LET P$=P$+"B"
2130   IF N=12 THEN LET P$=P$+"C"
2140   IF N=13 THEN LET P$=P$+"D"
2150   IF N=14 THEN LET P$=P$+"E"
2160   IF N=15 THEN LET P$=P$+"F"
2170 NEXT C
2180 RETURN
2190 REM
2200 REM *****
2210 REM * BUSQUEDA DE BYTES *

```

TRUCOS Y RUTINAS BASICAS

```

2220 REM *****
2230 REM
2240 GOSUB 4490
2250 INPUT "CUANTOS BUSCO? (1-5) ";B
2260 GOSUB 4490
2270 IF B>5 OR B<1 THEN GOTO 2240
2280 FOR A=1 TO B
2290   PRINT CHR$(147)
2300   POKE 214,21:POKE 211,0
2310   PRINT "DAME EL NUMERO ";A
2320   GOSUB 4490
2330   INPUT "NUMERO > ";C(A)
2340   GOSUB 4490
2350   IF C>255 OR C<0 THEN GOTO 2330
2360 NEXT A
2370 PRINT CHR$(147)
2380 GOSUB 4490
2390 PRINT "ESPERE UN MOMENTO. BUSCANDO"
2400 FOR I=0 TO 65535!
2410   LET NN=0
2420   FOR J=0 TO B-1
2430     IF PEEK(I+J)=C(J+1) THEN NN=NN+1
2440   NEXT J
2450   IF NN=B THEN LET A=I:GOTO 2470
2460 NEXT I
2470 LET MP=A
2480 PRINT CHR$(147)
2490 GOSUB 4490
2500 PRINT "ENCONTRADO. PULSE UNA TECLA"
2510 GET A$
2520 IF A$="" THEN GOTO 2510
2530 GOSUB 1570
2540 RETURN
2550 REM
2560 REM *****
2570 REM * SALTO A UNA DIRECCION DE MEMORIA *
2580 REM *****
2590 REM
2600 GOSUB 4490
2610 INPUT "A QUE DIRECCION SALTO? (0-65535) ";J
2620 GOSUB 4490
2630 IF J<0 OR J>65535! THEN GOTO 2600
2640 GOSUB 4490
2650 PRINT "ESTAS SEGURO? (S/N)"
2660 GET A$
2670 IF A$="" THEN GOTO 2660
2680 IF A$="S" OR A$="s" THEN SYS J
2690 IF A$<>"N" AND A$<>"n" THEN GOTO 2660
2700 GOTO 4150
2710 REM
2720 REM *****
2730 REM * CARGAR BYTES (LOAD) *
2740 REM *****
2750 REM
2760 GOSUB 2870
2770 IF A$="D" OR A$="d" THEN OPEN 3,8,15,D$:GOTO 2800
2780 IF A$<>"C" AND A$<>"c" THEN GOTO 2760
2790 OPEN 3,1,0,D$
2800 INPUT #3,NN
2810 FOR Z=0 TO NN-1
2820   INPUT #3,A
2830   POKE D+Z,A

```

```

2840 NEXT Z
2850 CLOSE 3
2860 RETURN
2870 REM
2880 REM *****
2890 REM * CASETE O DISCO? *
2900 REM *****
2910 REM
2920 GOSUB 4490
2930 PRINT "CASETE O DISCO (C/D) ?"
2940 GET A$
2950 IF A$="" THEN GOTO 2940
2960 GOSUB 4490
2970 INPUT "NOMBRE? (1-8 CHRS.) ";D$
2980 IF LEN(D$)>8 OR LEN(D$)=0 THEN GOTO 2960
2990 GOSUB 4490
3000 INPUT "DESPLAZAMIENTO ? ";D
3010 IF D<0 OR D>65535! THEN GOTO 2990
3020 GOSUB 4490
3030 RETURN
3040 REM
3050 REM *****
3060 REM * GRABAR BYTES (SAVE) *
3070 REM *****
3080 REM
3090 GOSUB 2870
3100 GOSUB 4490
3110 INPUT "LONGITUD? ";L
3120 IF L>85535! OR L<0 THEN GOTO 3100
3130 IF A$="D" OR A$="d" THEN OPEN 3,8,15,D$:GOTO 3160
3140 IF A$<>"C" AND A$<>"c" THEN GOTO 3090
3150 OPEN 3,1,1,D$
3160 PRINT #3,L
3170 FOR Z=0 TO L
3180 PRINT #3,PEEK(D+Z)
3190 NEXT Z
3200 RETURN
3210 REM
3220 REM *****
3230 REM * CAMBIAR PUNTERO DE MEMORIA *
3240 REM *****
3250 REM
3260 GOSUB 4490
3270 INPUT "MEMORY = ";M
3280 IF M>65535! OR M<0 THEN GOTO 3260
3290 GOSUB 4490
3300 LET MP=M
3310 RETURN
3320 REM
3330 REM *****
3340 REM * CAMBIO DE BASE *
3350 REM *****
3360 REM
3370 POKE 214,5:POKE 211,18:PRINT "1. BASE 2. "
3380 POKE 214,7:POKE 211,18:PRINT "2. BASE 10. "
3390 POKE 214,9:POKE 211,18:PRINT "3. BASE 16. "
3400 GET K$
3410 IF K$="" THEN GOTO 3400
3420 IF K$<>"1" AND K$<>"2" AND K$<>"3" THEN GOTO 3400
3430 LET BA=-(2 AND K$="1")-(10 AND K$="2")-(16 AND K$="3")
3440 LET MO=VAL(K$)
3450 LET K$=""

```

TRUCOS Y RUTINAS BASICAS

```

3460 RETURN
3470 REM
3480 REM *****
3490 REM * MUEVE BYTES *
3500 REM *****
3510 REM
3520 GOSUB 4490
3530 INPUT "DESDE? ";P
3540 IF P<0 OR P>65535! THEN GOTO 3520
3550 GOSUB 4490
3560 INPUT "HASTA? ";D
3570 IF D<0 OR D>65535! THEN GOTO 3550
3580 GOSUB 4490
3590 INPUT "LONGITUD? ";L
3600 IF L<0 OR L>65535! THEN GOTO 3580
3610 FOR I=0 TO L-1
3620     POKE D+L,PEEK(P+L)
3630 NEXT I
3640 RETURN
3650 REM
3660 REM *****
3670 REM * PASO DE DECIMAL A BINARIO *
3680 REM *****
3690 REM
3700 LET P$=""
3710 IF NU>=128 THEN LET P$="1":LET NU=NU-128:GOTO 3730
3720 P$="0"
3730 IF NU>=64 THEN LET P$=P$+"1":LET NU=NU-64:GOTO 3750
3740 P$=P$+"0"
3750 IF NU>=32 THEN LET P$=P$+"1":LET NU=NU-32:GOTO 3770
3760 P$=P$+"0"
3770 IF NU>=16 THEN LET P$=P$+"1":LET NU=NU-16:GOTO 3790
3780 P$=P$+"0"
3790 IF NU>=8 THEN LET P$=P$+"1":LET NU=NU-8:GOTO 3810
3800 P$=P$+"0"
3810 IF NU>=4 THEN LET P$=P$+"1":LET NU=NU-4:GOTO 3830
3820 P$=P$+"0"
3830 IF NU>=2 THEN LET P$=P$+"1":LET NU=NU-2:GOTO 3850
3840 P$=P$+"0"
3850 IF NU=1 THEN LET P$=P$+"1":LET NU=0:GOTO 3870
3860 P$=P$+"0"
3870 RETURN
3880 REM
3890 REM
3900 REM *****
3910 REM * INSTRUCCIONES *
3920 REM *****
3930 REM
3940 CLS
3950 PRINT "=====
3960 PRINT "** I N S T R U C C I O N E S **"
3970 PRINT "=====
3980 PRINT
3990 PRINT "D = DUMP. CAMBIA MODO."
4000 PRINT "B = BUSCA BYTES EN MEMORIA"
4010 PRINT "A = AYUDA. PONE ESTAS INSTRUCCIONES."
4020 PRINT "J = JUMP. EJECUTA CODIGO MAQUINA."
4030 PRINT "L = LOAD. CARGA BYTES."
4040 PRINT "M = MEMORY. AJUSTA EL PUNTERO."
4050 PRINT "P = PRINTER. PASA POR IMPRESORA."
4060 PRINT "S = SAVE. GRABA BYTES."

```

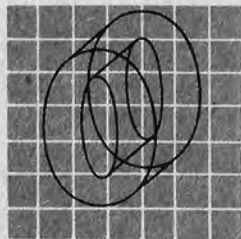
```

4070 PRINT "T = TRASLADA BYTES."
4080 PRINT "X = CAMBIA DE BASE."
4090 PRINT "2 = AUMENTA EL PUNTERO EN 10."
4100 PRINT "1 = AUMENTA EL PUNTERO EN UNO."
4110 PRINT "8 = DISMINUYE EL PUNTERO EN UNO."
4120 PRINT "8 = DISMINUYE EL PUNTERO EN 10."
4130 PRINT "0 = POKEA EN EL PUNTERO."
4140 PRINT
4150 PRINT "PULSA UNA TECLA"
4160 GET A$
4170 IF A$="" THEN GOTO 4180
4180 PRINT CHR$(147)
4190 RETURN
4200 REM
4210 REM *****
4220 REM * MODO DUMP DE MEMORIA *
4230 REM *****
4240 REM
4250 PRINT CHR$(147)
4260 FOR I=MP TO MP+150 STEP 10
4270   IF I>85535! THEN GOTO 4480
4280   LET N$=MID$(STR$(I),2)
4290   IF LEN(N$)<5 THEN LET N$="0"+N$:GOTO 4290
4300   PRINT#CH,N$;" ";
4310   LET Q$=""
4320   FOR J=I TO I+9
4330     LET NU=PEEK(J)
4340     LET M$=CHR$(NU)
4350     IF NU<32 THEN M$="."
4360     GOSUB 2020
4370     Q$=Q$+M$
4380     PRINT#CH,P$;
4390   NEXT J
4400   PRINT#CH," ";Q$
4410 NEXT I
4420 LET MP=MP+150
4430 GOSUB 4490:PRINT "CONTINUAMOS? (S/N)"
4440 GET A$
4450 IF A$="S" OR A$="s" THEN GOTO 4250
4460 IF A$<>"N" AND A$<>"n" THEN GOTO 4440
4470 GOSUB 4490
4480 RETURN
4490 REM
4500 REM *****
4510 REM * BORRADO DE LOA LINEA 22 Y POSICIONAMIENTO *
4520 REM *****
4530 REM
4540 POKE 214,22
4550 POKE 211,0
4560 FOR Z=1 TO 40
4570   PRINT " ";
4580 NEXT Z
4590 POKE 214,22
4600 POKE 211,0
4610 RETURN

```

EL TALLER DEL HARDWARE

■ Analizador lógico



TRA de las herramientas imprescindibles en el laboratorio de desarrollo de hardware es el analizador lógico. La circuitería necesaria para la captura de señales digitales es muy parecida a la descrita para el osciloscopio digital, con algunas variaciones para la generación de la señal de comienzo de la captura de señales y el programa de presentación y tratamiento de los datos. En este tomo describiremos los componentes básicos de un analizador lógico y su forma de empleo, utilizando un ordenador personal de tipo general a través de conexiones similares a la de tarjeta de ampliación de puertos de entrada/salida.

Las funciones básicas que realiza un analizador lógico son:

— Captura de las señales de interés de un sistema en instantes marcados por un reloj propio del analizador o del sistema.

— Comienzo del muestreo de las señales en el momento en que se produzca una circunstancia determinada y programable para cada ensayo.

— Representación de las señales almacenadas de forma gráfica o mediante caracteres, para determinación del comportamiento del sistema de la manera más eficiente posible.

— Almacenamiento de los resultados

de un experimento para su comparación con fuerce el circuito al que se conecta.

Todas estas funciones pueden realizarse con la capacidad lógica y gráfica de un ordenador personal, si se le añade una circuitería adicional para la adquisición de las señales a la velocidad oportuna.

En el análisis de sistemas digitales mediante analizador lógico suele interesar la determinación del comportamiento en dos aspectos:

— Análisis temporal, midiendo los tiempos de las diferentes señales en sus diferentes estados y sus relaciones con las demás. Se requiere un muestreo periódico a una velocidad suficiente para poder capturar los pulsos de mínima duración que puedan producirse. El reloj será normalmente asíncrono con el sistema.

— Análisis de estados, realizando el muestreo con un reloj propio del sistema bajo ensayo, para capturar las señales en los instantes de interés para el sistema. El reloj podrá ser derivado del propio del sistema, con algún desfase para poder ajustarse a las necesidades de los circuitos.

Veremos aplicación de los dos tipos de análisis en el montaje con ordenador personal.

■ Diagrama de bloques

El analizador lógico se compone de los siguientes bloques principales:

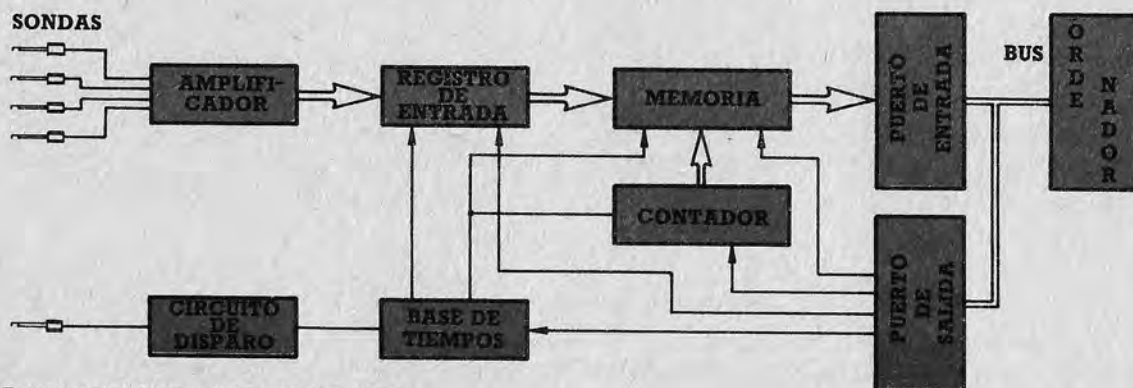


Fig. 1. Esquema de bloques del analizador lógico.

- Sonda de señales con amplificadores de adaptación.
- Registro de almacenamiento de entrada.
- Memoria de almacenamiento intermedio.
- Base de tiempos.
- Circuito de disparo.
- Conexión al ordenador personal.
- Panel de control.

Igual que en el ejemplo de osciloscopio digital, se ha optado por la solución de almacenamiento intermedio en una memoria externa, para independizar en lo posible la captura de las prestaciones del ordenador, sin por ello perder flexibilidad ni potencia de trabajo. Para algunas aplicaciones sería suficiente con la velocidad de transferencia de los puertos de entrada, pero es preferible montar un circuito que sea de aplicación general y a velocidades reales de trabajo.

■ Sonda de señales

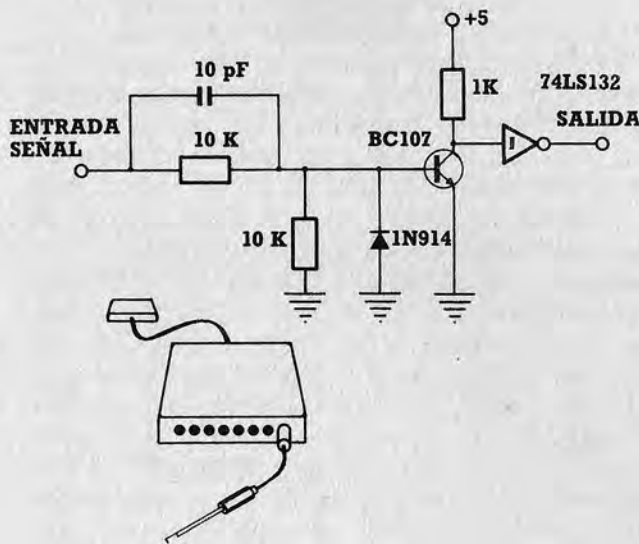


Fig. 2. Montaje de las sondas.

Es una parte importante del equipo, si pretendemos usarlo con frecuencia sin afectar al sistema que se ensaya. Constará cada entrada de señal de:

- Pinza miniatura para poder conectar a una patilla de circuito integrado o cualquier otro componente. Conviene seleccionarlás de colores para poder "seguirles la pista" al conectarlas a un sistema real. Conviene también

ponerles una etiqueta identificadora, indicando el número de la entrada correspondiente.

- Hilo flexible de conexión. Es conveniente adquirir hilo muy flexible para que no fuerce el circuito al que se conecta.

— Amplificador de entrada. Puede ser simplemente una puerta, si trabajamos siempre con los mismos niveles, aunque lo ideal sería una sonda de alta impedancia, para afectar lo mínimo posible al circuito que se ensaya. El amplificador puede ir en una caja separada y próxima al ensayo, uniéndose al ordenador personal a través de un cable apantallado activado por circuitos de amplificación.

— Conector de grupo. Cada conjunto de ocho hilos se conectarán en grupo al sistema mediante un conector. Así se podrá dejar un grupo de conexiones de forma permanente si se necesita cambiar a otro, pues el número de entradas será limitado.

■ Registro de almacenamiento de entrada

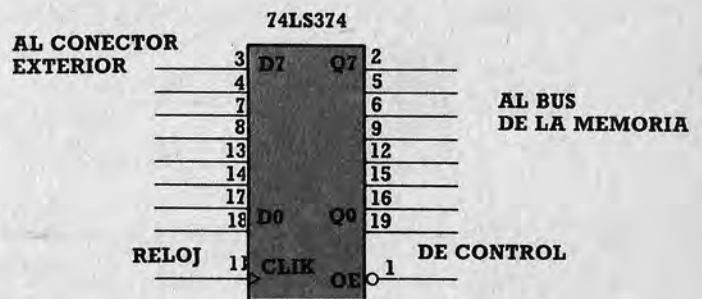


Fig. 3. Registro de almacenamiento de entrada.

Las señales se muestrean mediante un reloj, que determina el instante en el que se desea conocer su estado. El pulso del reloj produce la carga del registro de entrada con la información que en ese momento esté presente en sus entradas. Se emplea un circuito de ocho biestables de tipo 74LS374, para cada grupo de señales. La salida de estos biestables es de tipo de tres estados, para poder desactivarlos cuando la memoria es leída por el ordenador.

En modo adquisición la señal almacenada es mantenida entre dos pulsos de reloj, para que pueda ser cargada en la memoria de almacenamiento intermedio.

EL TALLER DEL HARDWARE

El número de registros de entrada puede ampliarse mediante circuitos idénticos al descrito, incluyendo además el circuito de memoria. El contador de direcciones puede ser el mismo para cada unidad de almacenamiento, siendo necesario realizar el control de la lectura mediante uno de los bits del puerto

de control, actuando sobre la entrada -CS de la memoria, para indicar cuál de ellas se desea leer.

Memoria de almacenamiento intermedio

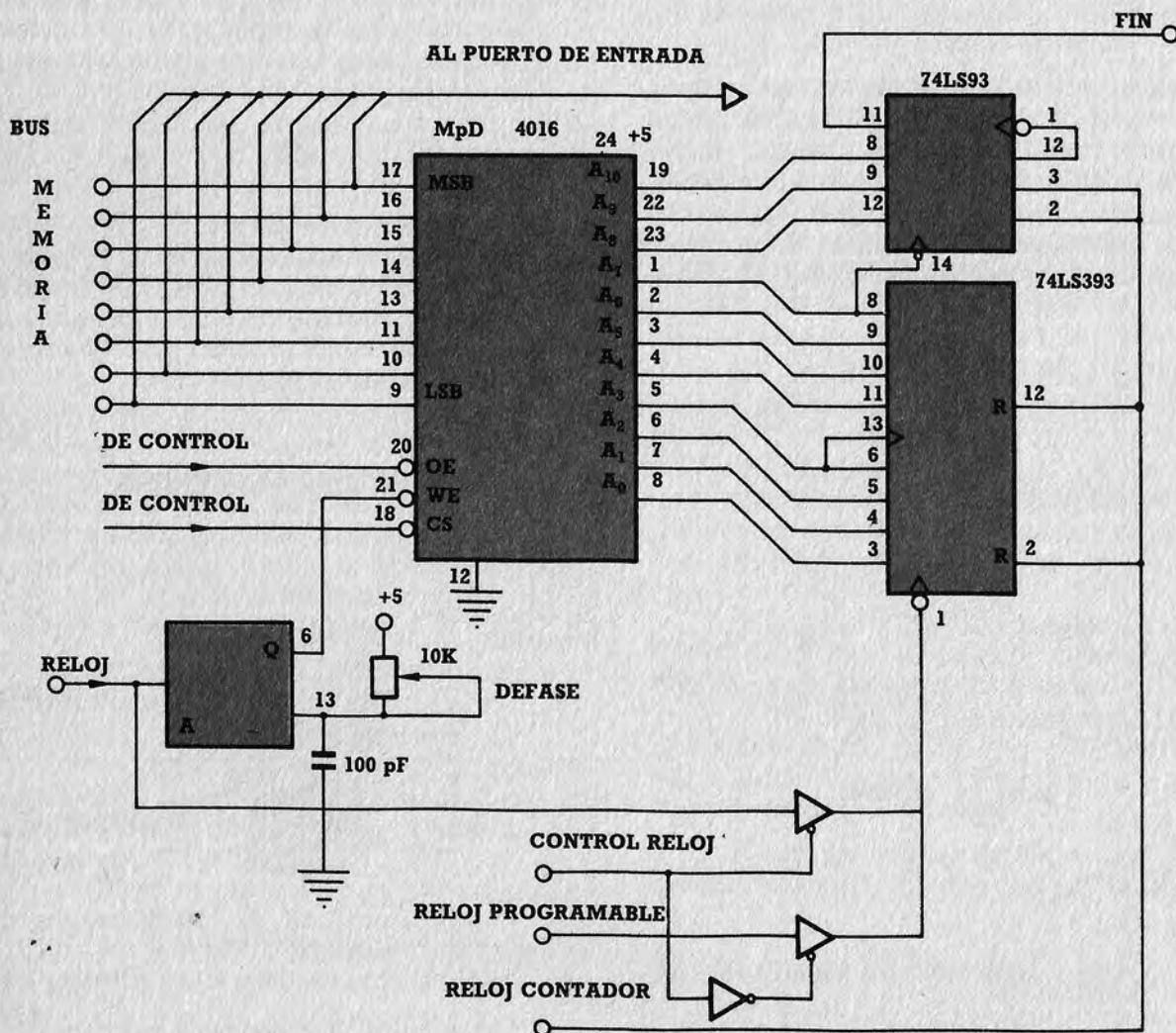


Fig. 4. Memoria de almacenamiento intermedio.

Unos pocos nanosegundos después de producirse la carga del registro de entrada están disponibles los datos para ser almacenados en la memoria, por lo que se genera un pulso de escritura. Mediante la interfaz con el ordenador se han activado los circuitos para que el pulso que se genere sea de escritura.

La memoria utilizada es de tipo estático, de 2K octetos, para cada grupo de entradas y el circuito se denomina 4016. La memo-

ria se escribe mediante el pulso -WE. Si esta entrada se deja en alto, a las salidas estarán disponibles los datos contenidos en la celda direccionada. Para que estén activas las salidas se activará la señal -OE, desde la interfaz.

El bus de direcciones de la memoria va conectado al contador de la base de tiempos. El bus de datos va conectado a las salidas del registro de entrada y las entradas del circuito de ampliación de puertos o similar.

■ Base de tiempos

muestreo normal. Interesa conocer si existen estas transiciones o "glitches", pues pueden in-

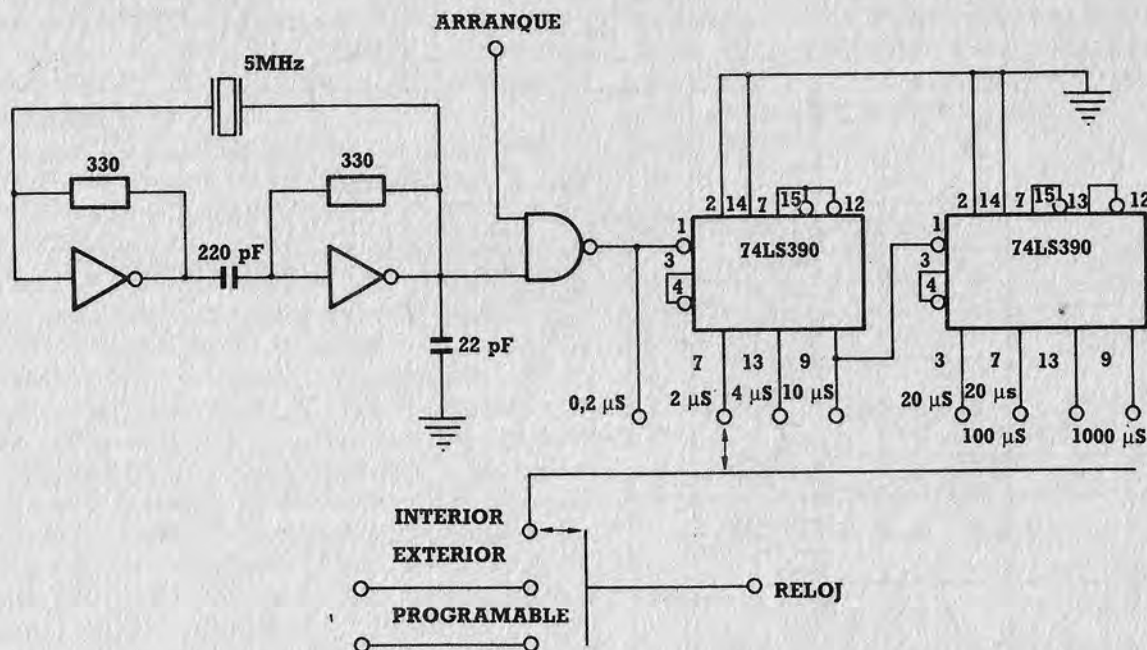


Fig. 5. Circuito de base de tiempos.

El circuito es similar al descrito para el osciloscopio digital, con el añadido de una entrada externa, para poder ser controlado por el propio reloj del sistema que se ensaya. También se ha añadido un circuito de retardo, para poder fijar un desfase entre la producción de la señal de reloj y la toma de la muestra.

El oscilador fundamental se compone de puertas inversoras y resistencias, que producen oscilación estable. Mediante un cristal de cuarzo en el lazo de la realimentación se consigue que la frecuencia de oscilación sea estable. La frecuencia más alta es de 5 MHz y puede seleccionarse la frecuencia de trabajo por escalones de 2 y 10 mediante un conmutador rotativo de 10 posiciones. La frecuencia máxima a la que, en realidad, podríamos muestrear está limitada por el tiempo de ciclo de la memoria utilizada. Si se dispone de una memoria más rápida podría modificarse el circuito oscilador para poder muestrear a una frecuencia igual a la inversa del tiempo de ciclo de la memoria.

El interés de muestrear a alta velocidad está en no perder muestras significativas de las señales a medir. En algunos equipos comerciales se dispone de circuitos que "observan" si existen transiciones entre dos muestras de la señal. Si estas transiciones son con subida y bajada pasarían desapercibidas en el

dicar la producción de un estado transitorio no deseado o la aparición de un ruido.

■ Circuito de disparo

Al analizar un sistema digital del tipo de un microcomputador las señales que se producen generan una cantidad enorme de información que para poder utilizarse prácticamente es necesario seleccionar. En general, deberemos comenzar el muestreo real de las señales cuando se produzca una señal de alerta indicativa del problema que pretendemos determinar o si no existen problemas, una secuencia que queremos conocer cómo se produce en condiciones reales.

Hay varias formas de definir las condiciones que indican el momento en que debemos comenzar a almacenar las señales:

- Al producirse una determinada combinación de bits en un bus. Puede ser una dirección o un conjunto de estados de diferentes puntos del sistema. Mediante un circuito de comparación determinamos el instante en que es necesario comenzar a muestrear.

- La condición puede ser una secuencia determinada de estados en uno o más pun-

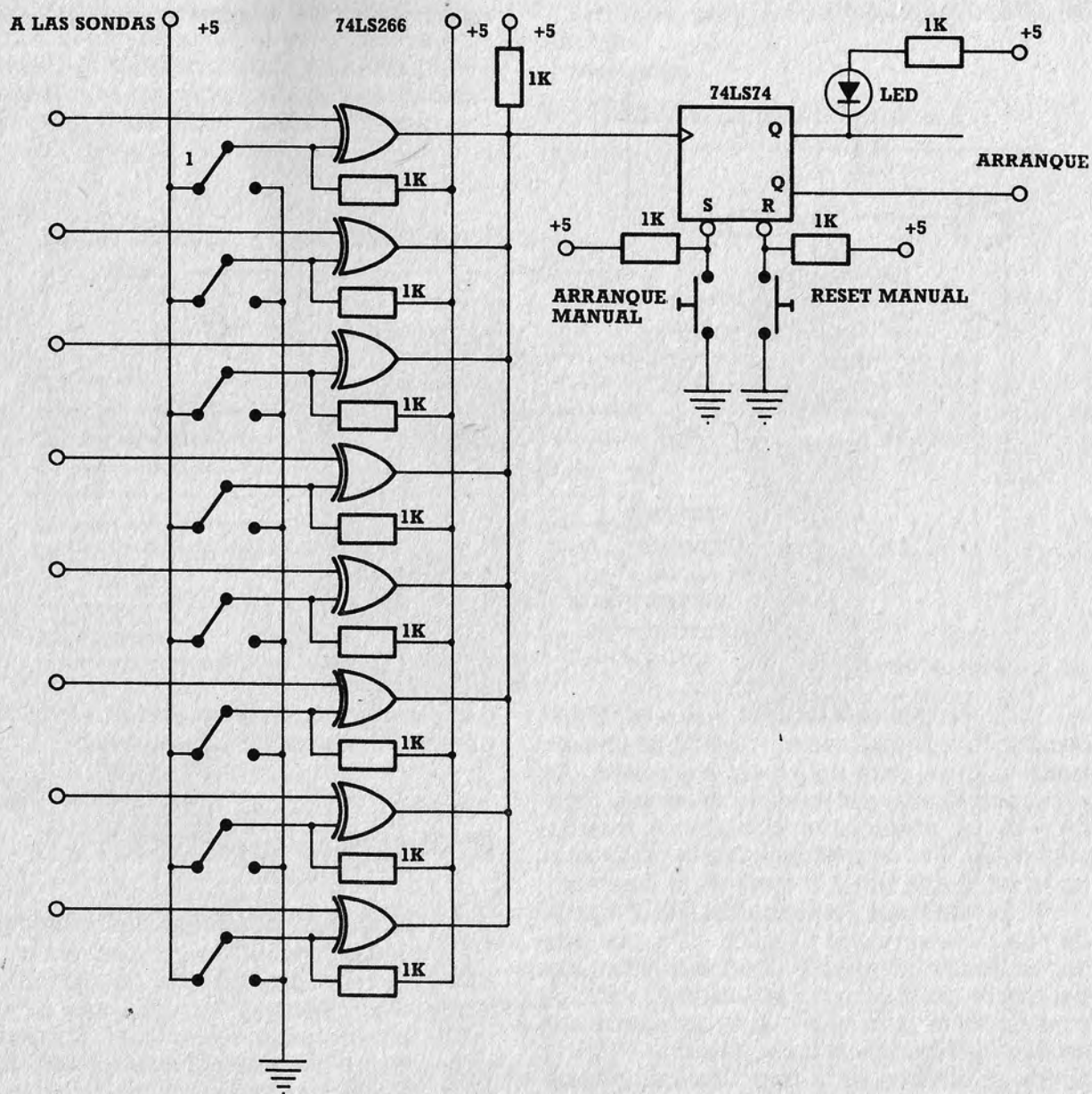


Fig. 6. Circuito de disparo por señales en paralelo.

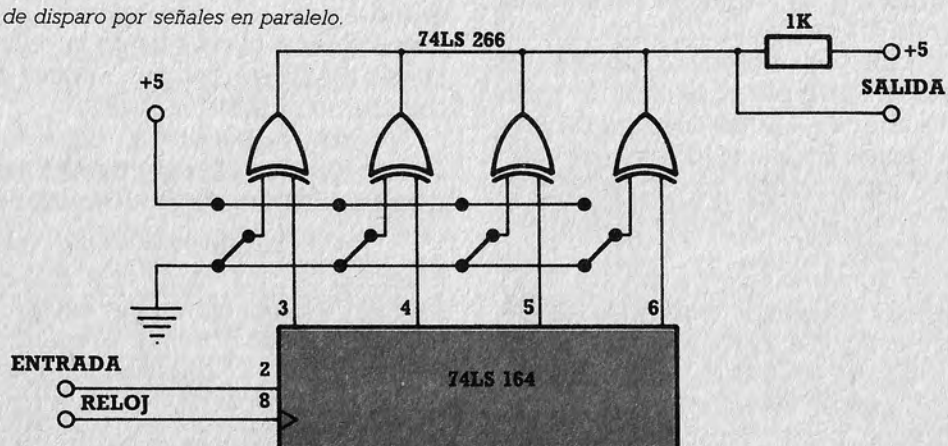


Fig. 7. Circuito de disparo por señales en serie.

tos del circuito. Necesitamos almacenar de forma serie las señales que determinan el disparo y mediante comparadores seleccionar la combinación que deseemos. Suele presentarse esta necesidad en el análisis de circuitos relacionados con equipos de transmisión de datos.

— Una forma mixta en la que intervienen varios puntos de forma paralela y la producción de una secuencia en serie.

En el circuito del ejemplo se muestra un tipo de circuito para cada uno de los casos y que puede seleccionarse por programa o por conmutadores desde el panel de mandos. El circuito básico es siempre un comparador realizado con puertas XOR (OR exclusivo) que dan una salida alta si las entradas están a niveles diferentes y una salida nula si las entradas están al mismo nivel, es decir, generan un 0 global si todas las entradas son iguales. En cada puerta comparadora se conecta una entrada al bus al que hay que utilizar para el disparo y la otra entrada a un conmutador que pueda conectarse a nivel 0 ó 1. La salida de las puertas es de tipo colector abierto.

Para la comparación con una señal serie se utiliza un registro de desplazamiento, activado con el reloj de muestreo de las señales, de tipo 74LS164.

■ Conexión al ordenador personal

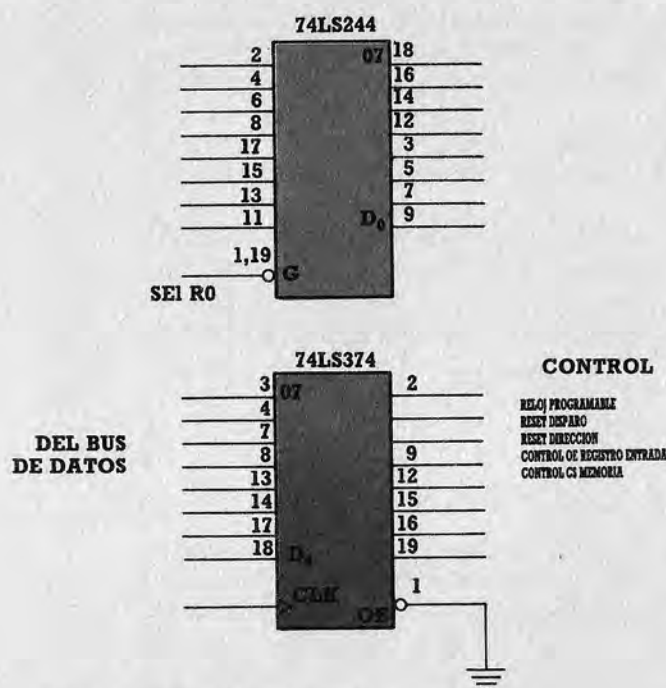


Fig. 8. Circuitos de interfaz con el ordenador.

Mediante la tarjeta de ampliación de puertos de entrada/salida, o con circuitos similares, se realizan las funciones de control, de indicación de estado y de transferencia de datos. La activación de los circuitos se efectúa mediante los circuitos de selección propios de cada ordenador personal, descrita en tomos anteriores.

■ Panel de control



Fig. 9. Panel de control.

Aunque todas las variables del sistema de captura de señales pueden controlarse desde el ordenador, puede resultar más práctico para nuestro caso el utilizar un pequeño panel de control que permita seleccionar la frecuencia de muestreo y las condiciones de disparo, además de mostrar el estado del muestreo mediante diodos luminosos.

La carátula se puede conectar directamente sobre el circuito impreso de tiras que contiene los circuitos.

■ Montaje práctico

El circuito resulta más simple que el del osciloscopio digital y además solamente contiene circuitos digitales. Es de suma importancia la conexión de las entradas de los puertos de adquisición de las señales, mediante conectores. Deberán disponer de dos patillas dedicadas a la masa común, por lo menos, para minimizar el efecto de los ruidos.

Los hilos de conexión al exterior para las sondas deberán ir apantallados para ser poco sensibles a las señales exteriores. Los hilos de las sondas deberán ser cortos hasta el amplificador de entrada. Si es necesario utilizar conexiones muy largas, porque así lo requiera el experimento, conviene que la conexión entre los amplificadores de las sondas

EL TALLER DEL HARDWARE

y los circuitos de entrada se haga con amplificadores (*drivers*) equilibrados, con ajuste de impedancias en el receptor, para reducir en lo posible el efecto de rebotes, que enmascararían las señales medidas.

■ Conclusión

El circuito descrito muestra, dentro de su sencillez, las innumerables posibilidades que ofrece a todo diseñador que posea un ordenador personal. Con él y el programa de control adecuado podremos capturar las señales de un sistema digital en condiciones de

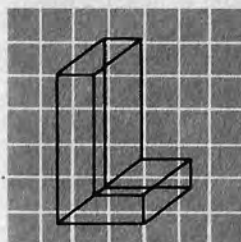
trabajo reales, sin interferir con su funcionamiento.

Como es lógico comprender, no todos los problemas están al alcance de un equipo como el presentado, pero la mayoría de los montajes de interés en nuestro taller e incluso el diagnóstico de problemas en periféricos pueden verse muy mejorados con la ayuda de una tarjeta como la presentada.

Los circuitos empleados son en la parte digital exactamente los mismos que los utilizados en el osciloscopio digital, por lo que para el experimentador que los hubiera adquirido será fácil adaptar el montaje para probar el nuevo equipo.

MATEMATICAS

■ Combinatoria



A *combinatoria* estudia de una forma general la formación y propiedades de ciertas agrupaciones de elementos, teniendo en cuenta o no el orden de colocación de dichos elementos y siguiendo una determinada

ley de formación.

En combinatoria se plantean dos problemas: formar las distintas agrupaciones posibles, de acuerdo con la ley de formación, y calcular el número total de agrupaciones distintas que se pueden formar. Este último tema es el objeto de nuestro programa.

Existen tres tipos de agrupaciones fundamentales: variaciones, permutaciones y combinaciones.

Llamamos *variaciones* de N elementos a cada una de las agrupaciones de A elementos diferentes que se pueden formar, de modo que dos agrupaciones difieran en algún elemento o en el orden de colocación de los mismos.

Para calcular el número de variaciones se utiliza la fórmula:

$$V_{N,A} = N * (N-1) * (N-2) * \dots * (N-A+1)$$

Las *variaciones con repetición* difieren de las ordinarias en que los A elementos que forman cada agrupación no tienen por qué ser diferentes.

Para calcular el número de variaciones con repetición utilizamos la fórmula:

$$VR_{N,A} = N^A$$

Se llaman permutaciones de N elementos a las distintas agrupaciones formadas por dichos N elementos, diferenciándose una agrupación de otra únicamente en el orden de colocación de sus elementos.

El número de permutaciones se calcula con la fórmula:

$$P = N! = 1 * 2 * 3 * \dots * N$$

Las *permutaciones* de N elementos divididos en grupos de *elementos repetidos* $A_1, A_2, \dots, A_i$ veces son las distintas agrupaciones que se pueden formar con los N elementos, diferenciándose una agrupación de otra en el orden de colocación de sus elementos.

El cálculo del número de permutaciones con repetición se efectúa con la fórmula:

$$PR_N^{A_1, A_2, \dots, A_i} = \frac{N!}{A_1! * A_2! * \dots * A_i!}$$

Finalmente, llamamos combinaciones de N elementos agrupados de A en A a cada una de estas agrupaciones, de forma que los A elementos sean diferentes y teniendo en cuenta que dos agrupaciones sólo son diferentes si difieren en algún elemento.

El cálculo del número de combinaciones se realiza con la fórmula:

$$C_{N,A} = \frac{V_{N,A}}{P_A}$$

El programa que vamos a ver a continuación nos permite resolver todos los problemas planteados anteriormente. Cada uno se estructura en una subrutina diferente.

```

10 REM *****
20 REM * COMBINATORIA *
30 REM *****
40 CLS
50 PRINT :PRINT
60 PRINT TAB(12);"OPCIONES"
70 PRINT TAB(17);"_____"
80 PRINT :PRINT
90 PRINT TAB(6);"1. VARIACIONES"
100 PRINT
110 PRINT TAB(6);"2. VARIACIONES CON REPETICION"
120 PRINT
130 PRINT TAB(6);"3. PERMUTACIONES"
140 PRINT
150 PRINT TAB(6);"4. PERMUTACIONES CON REPETICION"
160 PRINT
170 PRINT TAB(6);"5. COMBINACIONES"
180 PRINT :PRINT :PRINT
190 PRINT TAB(9);"PULSA LA OPCION DESEADA"
200 LET O$=INKEY$:IF O$="" THEN GOTO 200
210 IF ASC(O$)<49 OR ASC(O$)>53 THEN GOTO 200
220 CLS
230 LET OP=VAL(O$)
240 ON OP GOSUB 1000,2000,3000,4000,5000
250 CLS
260 PRINT "¿QUIERES CAMBIAR DE OPCION? (S/N)"
270 LET R$=INKEY$:IF R$="" THEN GOTO 270
280 IF R$="S" OR R$="s" THEN GOTO 40
290 END
1000 REM *****
1010 REM * VARIACIONES *
1020 REM *****
1030 INPUT "NUMERO DE ELEMENTOS";N:IF N<2 THEN GOTO 1030
1040 INPUT "AGRUPACIONES";A:IF A<1 OR A>N THEN GOTO 1040
1050 CLS
1060 LET V=1
1070 FOR I=N-A+1 TO N
1080 LET V=V*I
1090 NEXT I
1100 PRINT :PRINT
1110 PRINT "VARIACIONES DE ";N;" ELEMENTOS"
1120 PRINT "TOMADOS DE ";A;" EN ";A;" : "
1130 PRINT
1140 PRINT "V(";N;";";A;") = ";V
1150 PRINT :PRINT
1160 PRINT "¿QUIERES CALCULAR OTRA VARIACION? (S/N)"
1170 LET R$=INKEY$:IF R$="" THEN GOTO 1170
1180 IF R$="S" OR R$="s" THEN CLS:GOTO 1030
1190 RETURN
2000 REM *****
2010 REM * VARIACIONES CON REPETICION *
2020 REM *****
2030 INPUT "NUMERO DE ELEMENTOS";N:IF N<2 THEN GOTO 2030
2040 INPUT "AGRUPACIONES";A:IF A<1 OR A>N THEN GOTO 2040
2050 CLS
2060 LET VR=N^A
2070 PRINT :PRINT
2080 PRINT "VARIACIONES CON REPETICION DE ";N
2090 PRINT "ELEMENTOS TOMADOS DE ";A;" EN ";A;" : "
2100 PRINT
2110 PRINT "VR(";N;";";A;") = ";VR
2120 PRINT :PRINT
2130 PRINT "¿QUIERES CALCULAR OTRA VARIACION CON REPETICION? (S/N)"
2140 LET R$=INKEY$:IF R$="" THEN GOTO 2140
2150 IF R$="S" OR R$="s" THEN CLS:GOTO 2030
2160 RETURN
3000 REM *****
3010 REM * PERMUTACIONES *
3020 REM *****
3030 INPUT "NUMERO DE ELEMENTOS";N:IF N<2 THEN GOTO 3030
3040 CLS
3050 LET P=1
3060 FOR I=1 TO N
3070 LET P=P*I
3080 NEXT I
3090 PRINT :PRINT
3100 PRINT "PERMUTACIONES DE ";N;" ELEMENTOS : "
3110 PRINT
3120 PRINT "P(";N;") = ";P
3130 PRINT :PRINT :PRINT
3140 PRINT "¿QUIERES CALCULAR OTRA PERMUTACION? (S/N)"
3150 LET R$=INKEY$:IF R$="" THEN GOTO 3150
    
```

```

3160 IF R$="S" OR R$="s" THEN CLS:GOTO 3030
3170 RETURN
4000 REM *****
4010 REM * PERMUTACIONES CON REPETICION *
4020 REM *****
4030 INPUT "NUMERO DE ELEMENTOS";N:IF N<2 THEN GOTO 4030
4040 INPUT "NUMERO DE GRUPOS";A:IF A<1 OR A>N THEN GOTO 4040
4050 DIM P(A):DIM F(A)
4060 PRINT
4070 LET S=0
4080 FOR I=1 TO A
4090 INPUT "NUMERO DE ELEMENTOS DE UN GRUPO";P(I)
4100 IF P(I)<1 OR P(I)>N THEN GOTO 4090
4110 LET S=S+P(I)
4120 IF S>N THEN CLS:GOTO 4070
4130 NEXT I
4140 IF S<>N THEN CLS:GOTO 4070
4150 CLS
4160 LET T=1
4170 FOR I=1 TO A
4180 LET F(I)=1
4190 FOR J=1 TO P(I)
4200 LET F(I)=F(I)*J
4210 NEXT J
4220 LET T=T*F(I)
4230 NEXT I
4240 LET D=1
4250 FOR I=1 TO N
4260 LET D=D*I
4270 NEXT I
4280 LET PR=D/T
4290 PRINT :PRINT
4300 PRINT "PERMUTACIONES CON REPETICION DE ";N
4310 PRINT "ELEMENTOS CON ";A;" GRUPOS DE "
4320 FOR I=1 TO A-1
4330 PRINT P(I);",";
4340 NEXT I
4350 PRINT "Y ";P(A);" ELEMENTOS RESPECTIVAMENTE : "
4360 PRINT
4370 PRINT "PR(";N;",";
4380 FOR I=1 TO A
4390 PRINT P(I);",";
4400 NEXT I
4410 PRINT ") = ";PR
4420 PRINT :PRINT :PRINT
4430 PRINT "¿QUIERES CALCULAR OTRA PERMUTACION CON REPETICION? (S/N)"
4440 LET R$=INKEY$:IF R$="" THEN GOTO 4440
4450 IF R$="S" OR R$="s" THEN ERASE P,F:CLS:GOTO 4030
4460 RETURN
5000 REM *****
5010 REM * COMBINACIONES *
5020 REM *****
5030 INPUT "NUMERO DE ELEMENTOS";N:IF N<2 THEN GOTO 5030
5040 INPUT "AGRUPACIONES";A:IF A<1 OR A>N THEN GOTO 5040
5050 CLS
5060 LET V=1
5070 FOR I=N-A+1 TO N
5080 LET V=V*I
5090 NEXT I
5100 LET P=1
5110 FOR I=1 TO A
5120 LET P=P*I
5130 NEXT I
5140 LET D=V/P
5150 PRINT :PRINT
5160 PRINT "COMBINACIONES DE ";N;" ELEMENTOS"
5170 PRINT "TOMADOS DE ";A;" EN ";A;" I"
5180 PRINT
5190 PRINT "C(";N;",";A;) = ";C
5200 PRINT :PRINT :PRINT
5210 PRINT "¿QUIERES CALCULAR OTRA COMBINACION? (S/N)"
5220 LET R$=INKEY$:IF R$="" THEN GOTO 5220
5230 IF R$="S" OR R$="s" THEN CLS:GOTO 5030
5240 RETURN

```

Modificaciones para otros ordenadores

SPECTRUM:

210 IF CODE(O\$) < 49 OR CODE(O\$) >

53 THEN GOTO 200
240 GOSUB OP*1000
290 GOTO 9999

APRENDER CON EL ORDENADOR

Disminuir el argumento de todos los TAB en 5 unidades.

COMMODORE:

Sustituir todos los CLS por PRINT CHR\$(147).

Sustituir todas las asignaciones tipo: LET R\$=INKEY\$ por GET R\$.

Al ejecutar el programa aparecerá en pantalla un menú de opciones como el de la figura 1.

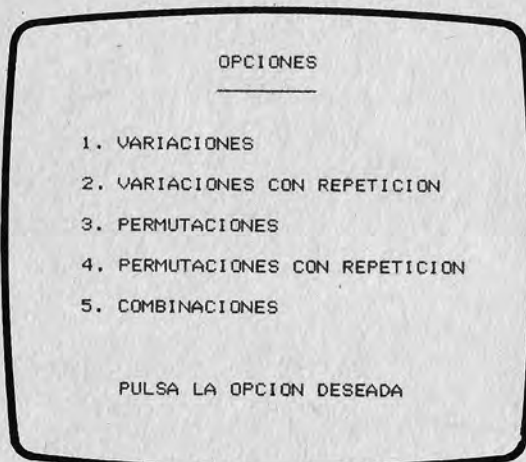


Fig. 1.

En la pantalla de la figura 2 podemos ver un ejemplo de aplicación concreta.



Fig. 2.

NATURALEZA Y TECNOLOGIA

■ Energía de un móvil en caída libre

La *energía* es la capacidad que tienen los cuerpos para realizar un trabajo.

La unidad de medida de la energía en el sistema internacional es el *julio*.

Hay dos formas principales de energía: energía cinética y energía potencial.

La energía cinética es la que poseen los cuerpos en movimiento producida por la aceleración causada por la acción de las fuerzas. Se calcula mediante la fórmula:

$$Ec = \frac{1}{2} m \cdot v^2$$

siendo *m* la masa del cuerpo y *v* su velocidad.

La energía potencial es el trabajo que puede producir un cuerpo y que tiene almacenado en forma latente. La energía potencial de un cuerpo situado a una altura *h* se calcula con la fórmula:

$$Ep = m \cdot g \cdot h$$

siendo *m* la masa del cuerpo y *g* la aceleración de la gravedad (9,8 m/s²).

Por otra parte, por el principio de conservación de la energía, se cumple que la suma de las energías potencial y cinética es siempre constante, independientemente de la situación del cuerpo.

$$Ep + Ec = Ep' + Ec'$$

Esta suma de energía potencial y cinética se denomina energía mecánica.

El programa que presentamos a continuación permite resolver distintos problemas relacionados con estas tres energías estudiadas, dependiendo de los datos de partida.

```

10 REM *****
20 REM * ENERGIA DE UN MOVIL EN CAIDA LIBRE *
30 REM *****
40 CLS
50 PRINT :PRINT
60 PRINT TAB(17);"OPCIONES"
70 PRINT TAB(17);"_____"
```

```

80 PRINT :PRINT
90 PRINT "1. E. POTENCIAL PARA UNA ALTURA DADA"
100 PRINT "2. E. CINETICA PARA UNA ALTURA DADA"
110 PRINT "3. E. MECANICA PARA UNA ALTURA DADA"
120 PRINT "4. VELOCIDAD PARA UNA ALTURA DADA"
130 PRINT "5. E. POTENCIAL PARA UNA VELOCIDAD DADA"
140 PRINT "6. E. CINETICA PARA UNA VELOCIDAD DADA"
150 PRINT "7. E. MECANICA PARA UNA VELOCIDAD DADA"
160 PRINT "8. ALTURA PARA UNA VELOCIDAD DADA"
170 PRINT :PRINT :PRINT
180 PRINT "PULSA LA OPCION DESEADA"
190 LET O$=INKEY$:IF O$="" THEN GOTO 190
200 IF ASC(O$)<49 OR ASC(O$)>56 THEN GOTO 190
210 CLS
220 LET OP=VAL(O$)
230 INPUT "MASA DEL CUERPO EN Kg.":M
240 IF M<=0 THEN GOTO 230
250 INPUT "ALTURA INICIAL DEL CUERPO EN m.":H
260 IF H<0 THEN GOTO 250
270 LET G=9.8
280 IF OP>=1 AND OP<=4 THEN GOTO 330
290 INPUT "VELOCIDAD DADA EN m/s":V1
300 LET V=SQR(2*G*H)
310 IF V1<0 OR V1>V THEN GOTO 290
320 GOTO 350
330 INPUT "ALTURA DADA EN m.":H1
340 IF H1<0 OR H1>H THEN GOTO 330
350 CLS
360 PRINT :PRINT
370 PRINT "MASA DEL CUERPO: ";M;" Kg"
380 PRINT
390 PRINT "ALTURA INICIAL DEL CUERPO: ";H;" m"
400 PRINT
410 IF OP>=1 AND OP<=4 THEN PRINT "ALTURA DADA: ";H1;" m":GOTO 430
420 PRINT "VELOCIDAD DADA: ";V1;" m/s"
430 PRINT
440 ON OP GOSUB 500,600,700,800,900,1000,1100,1200
450 PRINT :PRINT :PRINT
460 PRINT "QUIERES HACER OTRO CALCULO? (S/N)"
470 LET R$=INKEY$:IF R$="" THEN GOTO 470
480 IF R$="S" OR R$="s" THEN GOTO 40
490 END
500 REM *****
510 REM * ENERGIA POTENCIAL PARA UNA ALTURA DADA *
520 REM *****
530 LET EP=M*G*H1
540 PRINT "ENERGIA POTENCIAL: ";EP;" Julios"
550 RETURN
600 REM *****
610 REM * ENERGIA CINETICA PARA UNA ALTURA DADA *
620 REM *****
630 LET EC=M*G*(H-H1)
640 PRINT "ENERGIA CINETICA: ";EC;" Julios"
650 RETURN
700 REM *****
710 REM * ENERGIA MECANICA PARA UNA ALTURA DADA *
720 REM *****
730 LET EM=M*G*H
740 LET EP=M*G*H1
750 LET EC=EM-EP
760 PRINT "ENERGIA MECANICA: ";EM;" Julios"
770 PRINT :PRINT "ENERGIA POTENCIAL: ";EP;" Julios"
780 PRINT :PRINT "ENERGIA CINETICA: ";EC;" Julios"
790 RETURN
800 REM *****
810 REM * VELOCIDAD PARA UNA ALTURA DADA *
820 REM *****
830 LET V=SQR(2*G*(H-H1))
840 PRINT "VELOCIDAD: ";V;" m/s"
850 RETURN
900 REM *****
910 REM * ENERGIA POTENCIAL PARA UNA VELOCIDAD DADA *
920 REM *****
930 LET EP=M*G*H-M*V1*V1/2
940 PRINT "ENERGIA POTENCIAL: ";EP;" Julios"
950 RETURN
1000 REM *****
1010 REM * ENERGIA CINETICA PARA UNA VELOCIDAD DADA *
1020 REM *****
1030 LET EC=M*V1*V1/2
1040 PRINT "ENERGIA CINETICA: ";EC;" Julios"
1050 RETURN

```

APRENDER CON EL ORDENADOR

```

1100 REM *****
1110 REM * ENERGIA MECANICA PARA UNA VELOCIDAD DADA *
1120 REM *****
1130 LET EM=M*G*H
1140 LET EP=M*U1*U1/2
1150 LET EC=EM-EP
1160 PRINT "ENERGIA MECANICA: ";EM;" Julios"
1170 PRINT "ENERGIA POTENCIAL: ";EP;" Julios"
1180 PRINT "ENERGIA CINETICA: ";EC;" Julios"
1190 RETURN
1200 REM *****
1210 REM * ALTURA PARA UNA VELOCIDAD DADA *
1220 REM *****
1230 LET H1=H-(V1*V1/(2*G))
1240 PRINT "ALTURA: ";H1;" m"
1250 RETURN

```

Modificaciones para otros ordenadores

SPECTRUM:

```

200 IF CODE(0$) < 49 OR CODE(0$) >
56 THEN GOTO 190
440 GOSUB (OP+4)*100
490 GOTO 9999

```

Disminuir el argumento de todos los TAB en 5 unidades.

COMMODORE:

Sustituir todos los CLS por PRINT CHR\$(147).

Sustituir todas las asignaciones tipo LET R\$ = INKEY\$ por GET R\$.

En la pantalla del ordenador aparecen todas las opciones posibles:

```

          OPCIONES
          -----
1. E. POTENCIAL PARA UNA ALTURA DADA
2. E. CINETICA PARA UNA ALTURA DADA
3. E. MECANICA PARA UNA ALTURA DADA
4. VELOCIDAD PARA UNA ALTURA DADA
5. E. POTENCIAL PARA UNA VELOCIDAD DADA
6. E. CINETICA PARA UNA VELOCIDAD DADA
7. E. MECANICA PARA UNA VELOCIDAD DADA
8. ALTURA PARA UNA VELOCIDAD DADA

PULSA LA OPCION DESEADA

```

Fig. 3.

Una vez seleccionada la opción, el ordenador pide los datos necesarios y presenta en pantalla los resultados obtenidos.

```

MASA DEL CUERPO: 60 Kg
ALTURA INICIAL DEL CUERPO: 12 m
ALTURA DADA: 8 m
ENERGIA MECANICA: 7056 Julios
ENERGIA POTENCIAL: 4704 Julios
ENERGIA CINETICA: 2352 Julios

QUIERES HACER OTRO CALCULO? (S/N)

```

Fig. 4.

SOCIEDAD

Monedas del mundo

Hoy en día la economía y el comercio tienen una importancia fundamental. La gran mayoría de las transacciones comerciales tienen la moneda como base.

Sin embargo, es de todos sabido que cada país tiene su propia moneda.

El presente programa nos permite probar nuestros conocimientos en este tema. Para ello se plantea un test que puede seguir dos caminos, a elegir, inversos entre sí. En la primera opción el ordenador propone diez países elegidos al azar y para cada uno tres posibles monedas. En este caso tendremos que averiguar la moneda correspondiente al país en cuestión. La segunda opción plantea el test al contrario. En pantalla aparece una moneda y tres posibles países. Tenemos que acertar cuál de esos países corresponde a la moneda.

PAIS: JAPON

1. RUBLO
2. SCHILLING
3. YEN

CUAL ES LA RESPUESTA 1

INCORRECTO. LA RESPUESTA ES YEN

Fig. 5.

```
10 REM *****
20 REM *
30 REM * MONEDAS DEL MUNDO *
40 REM *
50 REM *****
60 CLS
70 DIM A$(30)
80 DIM B$(30)
90 DIM E(10)
110 PRINT :PRINT
120 PRINT "*****"
130 PRINT "*"
140 PRINT "TEST DE"
150 PRINT "*"
160 PRINT "*"
170 PRINT "MONEDAS DEL MUNDO"
180 PRINT "*"
190 PRINT "*****"
200 PRINT :PRINT :PRINT
210 PRINT "EL EXAMEN SE COMPONE DE DIEZ PREGUNTAS"
220 PRINT
230 PRINT "TIPO TEST CON TRES RESPUESTAS POSIBLES."
240 PRINT :PRINT
250 PRINT "DEBES PULSAR EL NUMERO DE"
260 PRINT
270 PRINT "LA RESPUESTA CORRECTA"
280 FOR T=1 TO 10000:NEXT T
290 REM *****
300 REM * SELECCION DEL TIPO DE TEST *
310 REM *****
320 CLS
330 PRINT :PRINT :PRINT
340 PRINT TAB(17);"OPCIONES"
350 PRINT TAB(17);"_____"
360 PRINT :PRINT :PRINT
370 PRINT TAB(7);"1. AVERIGUAR LA MONEDA"
380 PRINT TAB(7);"CORRESPONDIENTE A UN PAIS"
390 PRINT
400 PRINT TAB(7);"2. AVERIGUAR EL PAIS"
410 PRINT TAB(7);"CORRESPONDIENTE A UNA MONEDA"
420 PRINT :PRINT :PRINT
430 PRINT TAB(7);"PULSA LA OPCION DESEADA"
440 LET O$=INKEY$:IF O$="" THEN GOTO 440
450 IF O$<>"1" AND O$<>"2" THEN GOTO 440
460 CLS:RESTORE:LET C=0:LET A=0
470 REM *****
480 REM * CARGA DE LAS MATRICES *
490 REM *****
500 FOR I=1 TO 30
510 IF O$="1" THEN READ A$(I),B$(I):GOTO 530
520 READ B$(I),A$(I)
530 NEXT I
540 REM *****
550 REM * DIEZ PREGUNTAS ALEATORIAS *
560 REM *****
```

APRENDER CON EL ORDENADOR

```

570 RANDOMIZE TIMER
580 CLS
590 LET P=INT(RND(1)*30)+1
600 FOR I=1 TO 10
610 IF E(I)=P THEN GOTO 570
620 NEXT I
630 LET C=C+1
640 LET E(C)=P
650 PRINT :PRINT
660 IF C=1 THEN PRINT "PAIS: ";A$(P):GOTO 680
670 PRINT "MONEDA: ";A$(P)
680 PRINT :PRINT :PRINT
690 LET L=INT(RND(1)*3)+1
700 LET S=INT(RND(1)*30)+1
710 IF S=P THEN GOTO 700
720 LET T=INT(RND(1)*30)+1
730 IF T=P OR T=S THEN GOTO 720
740 IF L=1 THEN PRINT "1. ";B$(P):PRINT "2. ";B$(S):PRINT "3. ";B$(T)
750 IF L=2 THEN PRINT "1. ";B$(S):PRINT "2. ";B$(P):PRINT "3. ";B$(T)
760 IF L=3 THEN PRINT "1. ";B$(S):PRINT "2. ";B$(T):PRINT "3. ";B$(P)
770 PRINT :PRINT :PRINT
780 PRINT "CUAL ES LA RESPUESTA ";
790 LET R$=INKEY$:IF R$="" THEN GOTO 790
800 IF ASC(R$)<49 OR ASC(R$)>51 THEN GOTO 790
810 LET R=VAL(R$):PRINT R
820 PRINT :PRINT
830 IF R>L THEN PRINT "INCORRECTO. LA RESPUESTA ES ";B$(P):GOTO 860
840 PRINT "CORRECTO"
850 LET A=A+1
860 FOR T=1 TO 5000:NEXT T:IF C<10 THEN GOTO 570
870 REM *****
880 REM * RESULTADOS DEL TEST *
890 REM *****
900 CLS
910 PRINT :PRINT :PRINT
920 PRINT "RESULTADOS DEL TEST"
930 PRINT " "
940 PRINT :PRINT
950 PRINT "RESPUESTAS CORRECTAS: ";A
960 PRINT
970 PRINT "RESPUESTAS INCORRECTAS: ";10-A
980 PRINT :PRINT
990 PRINT "CALIFICACION: ";
1000 IF A<5 THEN PRINT "SUSPENSO"
1010 IF A=5 OR A=6 THEN PRINT "APROBADO"
1020 IF A=7 OR A=8 THEN PRINT "NOTABLE"
1030 IF A>8 THEN PRINT "SOBRESALIENTE"
1040 PRINT :PRINT :PRINT
1050 PRINT "QUIERES HACER OTRO TEST (S/N)";
1060 LET R$=INKEY$:IF R$="" THEN GOTO 1060
1070 IF R$="S" OR R$="s" THEN GOTO 320
1080 DATA ESPAÑA,PESETA,FRANCIA,FRANCO,REINO UNIDO,LIBRA,ITALIA,LIRA,PORTUGAL,ES
CUDO
1090 DATA GRECIA,DRACMA,JAPÓN,YEN,URSS,RUBLO,ALEMANIA,MARCO,SUECIA,CORONA
1100 DATA POLONIA,ZLOTY,INDIA,RUPIA,BOLIVIA,PESO,ECUADOR,SUCRE,ARGENTINA,AUSTRAL
1110 DATA ESTADOS UNIDOS,DÓLAR,AUSTRIA,SCHILLING,HOLANDA,FLORIN,BOTSWANA,PULA,BU
LGARIA,LEV
1120 DATA CHINA,RENMINBI,COSTA RICA,COLON,YUGOSLAVIA,DINAR,GUATEMALA,QUETZAL,PER
U,SOL
1130 DATA ISRAEL,SHEKEL,MARRUECOS,DIRHAM,COREA,WON,NIGERIA,NAIRA,ARABIA SAUDI,RI
YAL

```

Modificaciones para otros ordenadores

SPECTRUM:

```

70 DIM A$(30,15)
80 DIM B$(30,15)
570 RANDOMIZE 0
800 IF CODE(R$) < 49 OR CODE(R$) >
41 THEN GOTO 790

```

Disminuir el argumento de todos los TAB en 5 unidades.

Los datos de las líneas DATA deben ir entre comillas.

La función RND no lleva argumento.

COMMODORE:

Sustituir todos los CLS por PRINT CHR\$(147).

Sustituir todas las asignaciones tipo LET
 R\$ = INKEY\$ por GET R\$.
 Suprimir la línea 570.

MSX:

Suprimir la línea 570.

AMSTRAD:

570 RANDOMIZE TIME

Al finalizar el test del ordenador imprime en pantalla los resultados obtenidos.

PARA LOS MAS JOVENES

■ Dibujos geométricos

Este programa ha sido pensado como una introducción a la geometría a través de figuras geométricas regulares.

Podemos elegir entre tres opciones: polígonos, estrellas y flores de circunferencias.

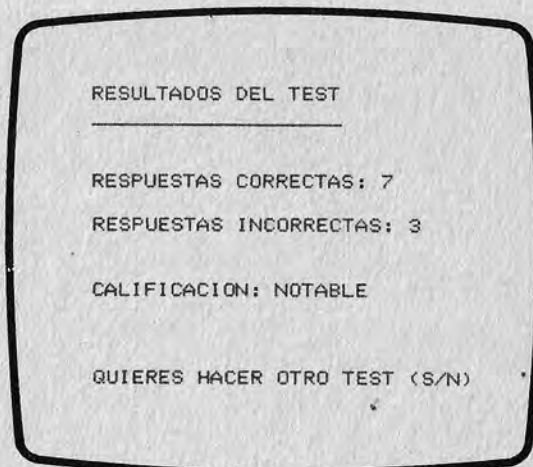


Fig. 6.

Tras elegir la opción el ordenador nos pide los datos necesarios para trazar la figura correspondiente, de modo que podemos comprobar la relación existente entre datos y dibujo resultante.

```

10 REM *****
20 REM *
30 REM *  DIBUJOS GEOMETRICOS  *
40 REM *****
50 CLS:SCREEN 1
70 PRINT :PRINT :PRINT
80 PRINT "*****"
90 PRINT " * "
100 PRINT " *      DIBUJOS      * "
110 PRINT " * "
120 PRINT " * "
130 PRINT " *      GEOMETRICOS    * "
140 PRINT " * "
150 PRINT " *****"
160 FOR T=1 TO 5000:NEXT T
170 REM *****
180 REM *  SELECCION DE OPCIONES  *
190 REM *****
200 CLS
210 PRINT :PRINT
220 PRINT TAB(17);"OPCIONES"
230 PRINT TAB(17);"_____"
240 PRINT :PRINT
250 PRINT TAB(15);"1. POLIGONOS"
260 PRINT
270 PRINT TAB(15);"2. ESTRELLAS"
280 PRINT
290 PRINT TAB(15);"3. FLORES"
300 PRINT
310 PRINT
320 PRINT
330 PRINT TAB(10);"PULSA LA OPCION DESEADA"
340 LET O$=INKEY$:IF O$="" THEN GOTO 340
350 IF ASC(O$)<49 OR ASC(O$)>51 THEN GOTO 340
360 CLS
370 LET OP=VAL(O$)
380 ON OP GOSUB 1000,2000,3000
390 CLS
400 PRINT "QUIERES CAMBIAR DE OPCION (S/N)"
410 LET R$=INKEY$:IF R$="" THEN GOTO 410
420 IF R$="S" OR R$="s" THEN GOTO 200
430 END
1000 REM *****
1010 REM *  POLIGONOS REGULARES  *
1020 REM *****
1030 INPUT "NUMERO DE LADOS";N:IF N<3 THEN GOTO 1030

```

APRENDER CON EL ORDENADOR

```

1040 INPUT "RADIO";R:IF R<5 OR R>90 THEN GOTO 1040
1050 CLS:LET C1=160:LET C2=100
1060 LET X1=C1+R
1070 LET Y1=C2
1080 LET A=2*3.1416/N
1090 PSET (X1,Y1)
1100 FOR I=A TO 2*3.1416+A STEP A
1110 LET X=C1+R*COS(I)
1120 LET Y=C2+R*SIN(I)
1130 LINE -(X,Y)
1140 LET X1=X:LET Y1=Y
1150 NEXT I
1160 PRINT "QUIERES PROBAR OTRO POLIGONO (S/N)"
1170 LET R$=INKEY$:IF R$="" THEN GOTO 1170
1180 IF R$="S" OR R$="s" THEN CLS:GOTO 1030
1190 RETURN
2000 REM *****
2010 REM * ESTRELLAS *
2020 REM *****
2030 INPUT "NUMERO DE PUNTAS";N:IF N<3 THEN GOTO 2030
2040 INPUT "RADIO MAYOR";R1:IF R1<5 OR 1>90 THEN GOTO 2040
2050 INPUT "RADIO MENOR";R2:IF R2<5 OR 2>R1 THEN GOTO 2050
2060 CLS:LET C1=160:LET C2=100
2070 LET X1=C1+R1
2080 LET Y1=C2
2090 LET A=3.1416/N:LET AN=0
2100 PSET (X1,Y1)
2110 FOR I=1 TO N
2120 LET AN=AN+A
2130 LET X=C1+R2*COS(AN)
2140 LET Y=C2+R2*SIN(AN)
2150 LINE -(X,Y)
2160 LET AN=AN+A
2170 LET X1=X:LET Y1=Y
2180 LET X=C1+R1*COS(AN)
2190 LET Y=C2+R1*SIN(AN)
2200 LINE -(X,Y)
2210 LET X1=X:LET Y1=Y
2220 NEXT I
2230 PRINT "QUIERES PROBAR OTRA ESTRELLA (S/N)"
2240 LET R$=INKEY$:IF R$="" THEN GOTO 2240
2250 IF R$="S" OR R$="s" THEN CLS:GOTO 2030
2260 RETURN
3000 REM *****
3010 REM * FLORES DE CIRCUNFERENCIAS *
3020 REM *****
3030 INPUT "NUMERO DE PETALOS";N:IF N<3 THEN GOTO 3030
3040 INPUT "RADIO PRINCIPAL";R1:IF R1<5 OR R1>85 THEN GOTO 3040
3050 INPUT "RADIO DE LOS PETALOS";R2:IF R2<5 OR R2>90-R1 THEN GOTO 3050
3060 CLS:LET C1=160:LET C2=100
3070 LET A=2*3.1416/N
3080 AN=0
3090 FOR I=1 TO N
3100 X=C1+R1*COS(AN)
3110 Y=C2+R1*SIN(AN)
3120 CIRCLE (X,Y),R2
3130 LET AN=AN+A
3140 NEXT I
3150 PRINT "QUIERES PROBAR OTRA FLOR (S/N)"
3160 LET R$=INKEY$:IF R$="" THEN GOTO 3160
3170 IF R$="S" OR R$="s" THEN CLS:GOTO 3030
3180 RETURN

```

Modificaciones para otros ordenadores

SPECTRUM:

```

350 IF CODE(O$) < 49 OR CODE(O$) >
51 THEN GOTO 340
380 GOSUB OP*1000
430 GOTO 9999
1050 CLS: LET C1=128: LET C2=88
1090 PLOT X1,Y1
1130 DRAW X-X1, Y-Y1

```

```

2060 CLS:LET C1=128: LET C2=88
2100 PLOT X1,Y1

```

```
2150 DRAW X-X1, Y-Y1
```

```
2200 DRAW X-X1, Y-Y1
```

```
3060 CLS: LET C1=128: LET C2=88
```

Suprimir en la línea 60 la instrucción
SCREEN 1.

AMSTRAD:

```
1050 CLS: LET C1=320: LET C2=200:
```

```

CLG
1090 PLOT X1,Y1
1130 DRAW X,Y
2060 CLS: LET C1=320: LET C2=200:
CLG
2100 PLOT X1,Y1
2150 DRAW X,Y
2200 DRAW X,Y
3060 CLS: LET C1=320: LET C2=200:
CLG

```

Sustituir la línea 3120 por la siguiente rutina:

```

3120 ORIGIN X,Y
3122 MOVE R2,0
3124 FOR B=0 TO 2*3,1416 STEP
3,1416/180
3126 DRAW R2*COS(B), R2*SIN(B)
3128 NEXT B

```

Sustituir en la línea 60 la instrucción SCREEN 1 por MODE 1.

MSX:

```

1050 CLS: LET C1=128: LET C2=96
2060 CLS: LET C1=128: LET C2=96

```

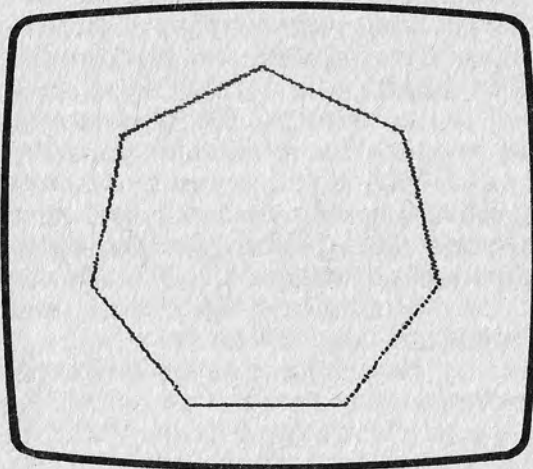


Fig. 9.

```

3060 CLS: LET C1=128: LET C2=96
Sustituir en la línea 60 la instrucción
SCREEN 1 por SCREEN 2.

```

Para finalizar, algunas muestras de los resultados que se pueden obtener con este programa.

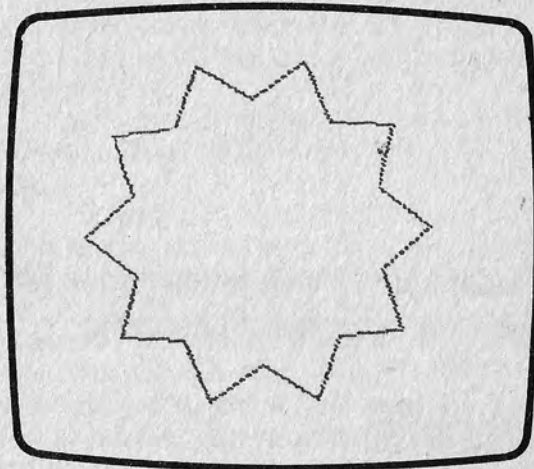


Fig. 7.

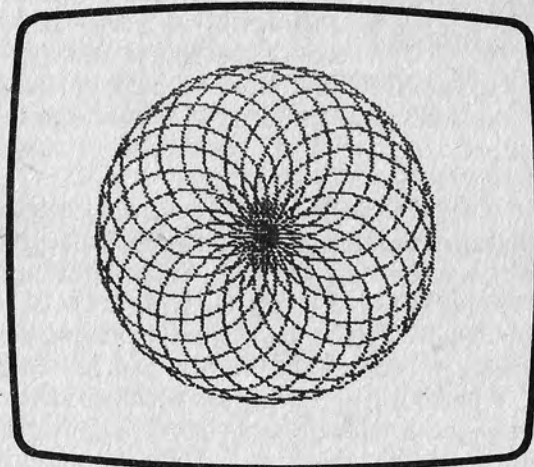
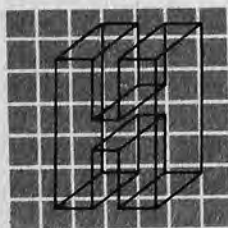


Fig. 8.

PEQUEÑA HISTORIA DE LA INFORMATICA

■ Ordenadores y otras máquinas IBM de los años cuarenta y cincuenta



ASTA ahora no hemos hablado mucho de la IBM. ¿Qué pasó con la gran compañía? ¿Se quedó a la zaga? Efectivamente (los años cuarenta de los que estábamos hablando) la IBM es una gran empresa, pero no se ha

atrevido por el momento a arriesgar en investigación, a apostar por las novedades que iban cambiando las grandes máquinas calculadoras universales.

Thomas Watson Jr. (del que tenemos que hablar más detenidamente) había entrado en el negocio de su padre. La compañía había tenido bastante éxito con el modelo de calculador 603, y Watson Jr. quería modernizar la compañía y lanzarse a invertir en investigación. Y pidió a R. Palmer que formara un grupo para desarrollar una máquina superior a la 603. El modelo creado fue el 604, que aunque seguía siendo una máquina de tarjetas perforadas, de uso muy restringido, era muy veloz, comparada con los modelos anteriores. Utilizaba muchos tubos de vacío (unos 1.500) y tenía una memoria de 50 dígitos decimales, el doble que la del modelo 603. Su velocidad de cálculo era también mucho mayor. Dividía directamente (el modelo 603 no lo hacía) y podía realizar las operaciones elementales, ejecutando unas cien tarjetas por minuto. Además, realizaba repeticiones de ciertas partes del programa (sin demasiados problemas) a la hora de programar la máquina. Pero no eran las innovaciones técnicas de la máquina las únicas. También tenía innovaciones de tipo

práctico. Así, los circuitos se comenzaron a montar en tarjetas, fácilmente sustituibles, si ocurrían fallos en la máquina. Esta idea de que el valor del material no tenía importancia frente al tiempo de las personas, y de ofrecer por encima de todo un buen servicio fue revolucionaria por su modernidad. Y naturalmente, en poco tiempo fue copiada por la mayor parte de los fabricantes.

La máquina (el modelo 604) fue un éxito. En 1948, hace ya casi treinta años, se vendieron ¡5.000 máquinas!, cantidad increíble, dado su precio.

Ese mismo año, el 605 (muy semejante al 604) se unió a una máquina de calcular tradicional de IBM, la 402, con el fin de obtener mejores sistemas de entrada/salida, aumentar la memoria (hasta 480 dígitos decimales) y otras pequeñas ventajas. El resultado fue el CPC (Card Programmed Calculator), del que IBM vendió más de setecientos.

Pero los últimos años de la década de los cuarenta y los primeros de la de los cincuenta eran años de cambios. En el seno de la IBM se estaban formando grupos que necesitaban cambiar el rumbo, y fabricar ordenadores. A la cabeza de ellos estaba Thomas Watson Jr., serio, trabajador, mucho mejor preparado técnicamente que su padre, sobre todo más joven y moderno. Además, había sido piloto de guerra, y tenía muchos amigos interesados en tecnologías de vanguardia como las que se utilizan en aviación. Algunos de ellos habían ingresado en la IBM, e intentaban darle nuevos aires. Además de este grupo, existía otro muy voluminoso de seguidores de Thomas Watson padre, que estaban muy contentos con el negocio de las máquinas de oficina de tarjetas perforadas, calculadores, etc., y no encontraban interesante pasar a construir otro

tipo de máquinas. Además, si fabricaban ordenadores, ¿qué ocurriría con el enorme parque de máquinas IBM vendidas? Los ordenadores les harían la competencia.

Pero Watson Jr. era casi tan tozudo como su padre, y su «clan» ganó la batalla. Ofreció a Von Neumann, que estaba entonces en el Institute for Advanced Studies, en Princeton, un cargo de asesor, y se lanzó de lleno a crear otras máquinas que sacaran provecho de las características de las cintas magnéticas, de las que esperaba mucho.

Con este caldo de cultivo otras circunstancias externas ayudaron a Watson a fabricar el modelo 701, y entre ellas tuvo gran importancia la guerra de Corea. Como toda gran compañía, la IBM estaba muy bien relacionada. Y en ocasiones, para hacer negocio, ni siquiera tenía que molestarse. Dada su envergadura, los negocios le venían rodados. Bastaba con aceptarlos.

El Ejército de los Estados Unidos ha sido siempre un magnífico cliente. Además, la tecnología que utiliza es siempre de vanguardia. el Whirlwind (diseñado para la Marina-Ejército del Aire, según se mire) fue una máquina revolucionaria. Rapidísima, realizaba acciones que no podían ni soñarse en esos años. Sin embargo, su problema era la fiabilidad. La máquina se estropeaba demasiado, y ¿qué utilidad tenía alcanzar velocidades tan altas si toda la eficacia se perdía en los tiempos muertos en que la máquina era reparada? En los medios adecuados del Ejército se llegó a la conclusión de que la máquina era magnífica, pero que era necesario obtener otra similar, pero mucho más fiable, y fabricarla a nivel industrial. Estaba claro que, dados los tiempos que corrían, las artesanías se estaban haciendo cada vez más caras.

Y el Ejército de los Estados Unidos invitó a la IBM a analizar la máquina e intentar obtener otras semejantes para el Ejército (naturalmente, fabricadas a nivel industrial).

La compañía se interesó en el proyecto, tomando como consultor a Von Neumann, que gozaba de un prestigio a nivel internacional cada día más alto. Inspirados en la máquina IAS, nacieron dos ordenadores gemelos, que fueron comercializados, uno de ellos pensando en los usuarios militares (o de investigación en el campo militar), el 701, y otro, pensando en un uso civil, principalmente de gestión, el 702.

De los dos ordenadores, el 701 se pensó para trabajar utilizando sistema binario, y

el 702, su gemelo, en sistema decimal. Estaban contruidos con un modelo de tubos de vacío cuyas características eran bastante mejores que los primeros que aparecieron. En efecto, los tubos de Williams eran mucho menores, y disipaban menos potencia, durando bastante más. En cuanto a la velocidad de cálculo, el 701 era capaz de efectuar 2.000 multiplicaciones por segundo, o dieciséis mil sumas por segundo. El primer modelo se presentó en abril de 1953.

Echemos un rápido vistazo a los restantes ordenadores IBM de la década de los cincuenta.

El 704 es algo posterior a los anteriores, y fue comercializado en 1954, aunque no se pudo disponer de unidades hasta bastante más tarde. Su diseñador fue Gene Amdahl, verdadero artífice de la serie 360. El 704 se distinguía de las demás fundamentalmente por su memoria, que era enorme (32.768 palabras de 35 bits) en la versión de 1957. La memoria era de toros de ferrita, copiada prácticamente de la del Whirlwind, aunque mucho mayor. La velocidad de cómputo era muy superior a la del 701 (unas tres veces más rápida) pero su mayor ventaja consistía en que sólo se estropeaba cada ocho o diez días. Esta máquina fue la antecesora de la IBM 709, último calculador de IBM que se fabricó con tubos.

No olvidemos que uno de los mayores éxitos comerciales le llegó a IBM a mediados de la década de los cincuenta con una máquina, el 650, cuyas características no ofrecían nada espectacular si no es una memoria de tambor magnético. Que podía almacenar hasta 150.000 bits de información. El dispositivo consistía en una columna que giraba sobre su eje a una velocidad de 12.500 revoluciones por minuto, sobre la que se «escribía» la información. Posteriormente, fue dotada de bandas magnéticas, y de otra memoria de toros de ferrita. Sin embargo, la serie de 700 era muy distinta en muchos aspectos. Suponía el paso de la máquina dedicada al ordenador.

Hasta el momento no hemos hablado de compatibilidad. Sin embargo, a mediados de los cincuenta ya existen en el mundo numerosos fabricantes de ordenadores, que van fabricando modelos cada cierto tiempo, intentando superar a la competencia, e incluso a sí mismos. Hemos hablado del 702 de IBM (gemelo del 701), pero pensado para fines civiles. En 1954 apareció el 705, con un nuevo dispositivo, el TRC (Tape Recorder Coordinator), que

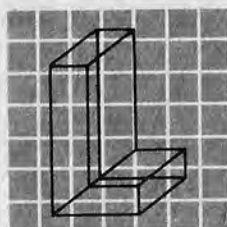
PEQUEÑA HISTORIA DE LA INFORMATICA

controlaba cintas. Su procesador central era bastante lento, pero la capacidad de su memoria, enorme, ya que con el sistema de cintas se podían almacenar hasta 20.000 caracteres en toros de ferrita (en los modelos últimos se llegaba a poder almacenar casi el doble). La memoria principal de los dos modelos era de tambor magnético, con una capacidad de 300.000 caracteres. ¿Podían, pues, servir las instrucciones y, en general, los programas del 702 para el 705? Evidentemente sería muy útil, pues las empresas y organismos que hubieran comprado un 702 podrían mejorar comprando un 705 y aprovechar todos los programas creados para su primera máquina. Sin embargo, esto no era posible, a pesar de la similitud entre las dos máquinas. Idéntico problema ocurría entre el modelo 704 y el 705. A pesar de ser dos modelos seguidos, eran bastante diferentes.

IBM se planteó en serio el problema de las incompatibilidades. Y a ello se debe que dispusiera del mismo equipo de entrada y de salida en las dos máquinas (704 y 705), a pesar de que existían otros problemas de incompatibilidad muy importantes. Desde principios de los años cincuenta, IBM repetía los dispositivos de entrada (lectora de tarjetas) y de salida (impresora), a pesar de que se habían realizado enormes avances, y podría haber colocado en sus equipos otros dispositivos más avanzados. Y así, el modelo 704 disponía de la misma lectora e impresora que el 701, sabiendo que eran lentas y anticuadas. Y el mismo problema se presentó para el modelo 705, aunque en este caso, al menos, hubo grandes deliberaciones, y todo tipo de opiniones. Sin embargo, tras las largas consideraciones triunfó el «sentido común» sobre el progreso, y a pesar de que ambas máquinas se habían diseñado independientemente, y que utilizaban códigos distintos para los caracteres alfabéticos, y otras «pequeñeces», IBM continuó con su antigua lectora e impresora. Pero fue el momen-

to en el que comenzó a interesarse por traducir el software de unas máquinas a otras, y sobre todo en comenzar a fabricar «series» de máquinas, en las que, al menos, los códigos básicos fueran compatibles. (Este problema fue agudizándose en la segunda generación de ordenadores.) IBM fabricó toda una serie de ordenadores transistorizados, los 7070, que estaban basados en el 705, pero habían adoptado tecnología moderna. Sin embargo, la serie completa no tuvo éxito, y ello se debió probablemente a que la nueva serie era incompatible con el ordenador primitivo del que provenía. Tras el fracaso, a la serie 7070 le siguió la 7080, que sí era compatible con el 705. Esta nueva serie fue un éxito, y demostró la importancia que tenía (y tiene) no perder un cliente. Una vez que «entra» en la casa, no hay que dejarlo salir, y debe seguir comprando productos cada vez mejores, confiando en su vendedor (una especie de confesor) que va asesorando al tímido y pequeño cliente, que es lo que debe ir haciendo a continuación. (Naturalmente, el cliente debe sentirse cuidado, incluso «mimado», por su vendedor-confesor, para no tener tentaciones de salir de la cómoda casa en la que ha entrado. Al fin y al cabo, ¿para qué complicarse la vida? Todo lo que hemos expuesto es válido para todas las grandes casas fabricantes de ordenadores, sin excepción, aunque algunas hayan superado a las demás en este punto. Todas ellas tenían un halo de «ocultismo científico» muy útil a sus intereses, además de fuerza económica para responder. Los ordenadores eran tan caros que sólo empresas muy potentes económicamente y bien relacionadas podían invertir cantidades enormes en investigación.

Esto justifica el interés demostrado por las compañías en cuidar la continuidad de las series de máquinas: son muchos los millones de dólares que ha reportado a las casas fabricantes cuidar la compatibilidad de sus máquinas.



A ingeniería del conocimiento es la rama de la Ingeniería Artificial que se ocupa precisamente de los modos de representación del conocimiento (los datos, las informaciones) y de su adecuado procesamiento.

Los resultados prácticos del trabajo de los «ingenieros del conocimiento» son los «sistemas basados en el conocimiento» o «sistemas expertos».

Normalmente está indicada la creación de un sistema basado en el conocimiento cuando el proceso no está bien definido o la información a manejar en cada toma de decisión es muy variada.

En estas condiciones, se incluyen en el sistema no sólo las reglas de producción propiamente dichas, sino reglas que aportan «estrategias» de búsqueda o reglas de elección (que se llaman «heurísticas»); estas reglas incluyen información del tipo «Si el paciente tiene fiebre alta, ENTONCES empezar examinando las reglas que consideran las características de los cultivos de laboratorio», que en la parte de la acción indica qué reglas han de ser utilizadas en primer lugar para el establecimiento del diagnóstico.

Es importante observar, además, que estas «metarreglas» («reglas de manejo de reglas») hacen referencia a las reglas primarias de la base de conocimientos por su contenido (en el caso anterior se indica aplicar ciertas reglas del diagnóstico que aluden a unas pruebas concretas) en vez de aludir a ellas por su nombre, como se hace en la programación convencional (alusión a las «etiquetas» de las instrucciones o nombres de los procedimientos).

Esto hace que no haya que reprogramar si se añade una nueva regla y, por tanto, que el sistema pueda ir incrementando automáticamente su propio acervo de conocimientos: el sistema «aprende».

De un modo un tanto simplista, se puede decir que si el sistema incluye una gran base de datos y un «pequeño» algoritmo de tratamiento de datos (los clásicos procesos repetitivos), el esquema óptimo es el de un proceso convencional de tipo «procedural»; si por el contrario, la base de hechos a considerar es pequeña (en un sistema de diagnóstico sólo los síntomas), pero la información sobre el proceso es amplia y no bien estructurada (una gran base de reglas alternativas o de aplicación parcial y condicionada) el esquema a desarrollar es el de un sistema basado en el conocimiento. Una de las ventajas básicas que aporta un sistema de este tipo es la flexibilidad y, de hecho, el nivel de eficacia del sistema («performance») viene dado más por la bondad de las metarreglas que se incluyen (para la búsqueda inteligente de las estrategias de proceso) que por las propias reglas de proceso que incorpore.

Por otro lado, es significativo constatar que los sistemas basados en el conocimiento son mantenidos directamente por los expertos, con la colaboración puntual del ingeniero del conocimiento. Esto se debe, entre otras cosas, al hecho de que la propia base de conocimientos de un sistema experto es directamente legible por un no-informático (son reglas del tipo de las anteriormente mencionadas que, en ocasiones, ha escrito directamente el experto —no el ingeniero del conocimiento—); además la propia estructura del sistema hace que sean fácilmente modificables y ampliables por el experto.

Otra característica sumamente interesante de los sistemas basados en el conocimiento es que la estructura básica de ellos viene dada fundamentalmente por las reglas de tipo heurístico que incorporan, mientras que un programa convencional se vincula a un algoritmo y de él obtiene la estructura completa del proceso a realizar.

Es decir, una de las claves de los sistemas basados en el conocimiento es precisamente la eficiencia de la base de conocimientos. La eficiencia, por otro lado, de dicha base viene dada por la acertada elección del esquema de representación a utilizar.

Al hablar del conocimiento incluido en una base de conocimiento es necesario distinguir entre las informaciones factuales (acerca de «hechos» y las reglas deductivas (aportan información acerca de cómo obtener nuevos datos o «hechos» a partir de los que se dispone).

De otro lado, los diferentes procedimientos disponibles para representar el conocimiento suelen ir vinculados a diferentes lenguajes o sistemas de representación, con lo que es necesario conocer las cualidades implícitas al sistema elegido para hacer un eficaz uso de ellas.

Uno de los procedimientos básicos de representar el conocimiento es el de las RE-

DES SEMANTICAS, de donde deriva las «TERNAS objeto-atributo-valor» y las «PAREJAS atributo-valor», así como «en cierto modo» los MARCOS o «frames» y GUIONES o «scripts»; otro de los procedimientos básicos es el proporcionado por las EXPRESIONES LOGICAS (cálculo de proposiciones y lógica de primer orden). Veamos en qué consisten cada uno de estos procedimientos.

■ Redes semánticas

Es un esquema genérico implementado en numerosos lenguajes y herramientas. Originalmente fue propuesto para el análisis semántico, pero la simplicidad relativa con que se pueden obtener deducciones correctas una vez ha sido generada la red es, sin duda, una de las razones básicas que justifican su enorme desarrollo actual (aunque, históricamente, es uno de los primeros esquemas de representación utilizados). La red está constituida por un conjunto de objetos o elementos llamados «nodos», conectados mediante unos «arcos» o «ligaduras». Tanto los arcos como los nodos son elementos básicos de la red y están etiquetados.

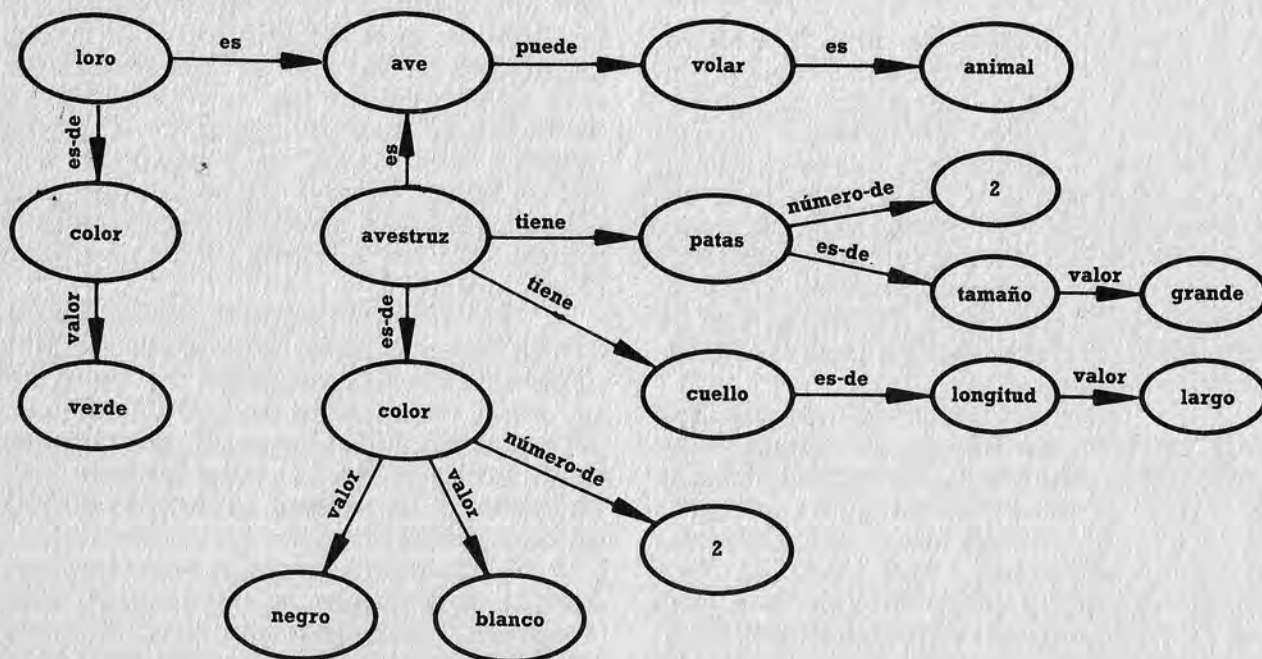


Fig. 1. Red semántica.

En principio no existen restricciones en el uso de las redes semánticas, los nodos pueden ser tanto objetos físicos (el avestruz, el cuello, las patas, etc.), como **categorías** abstractas (ave, animal...), **cualidades** definibles de los anteriores objetos o categorías (tamaño, longitud...), **valores** que toman las anteriores cualidades (grande el tamaño, verde el color...) e incluso **descriptores** de algún elemento (dos como número de patas, volador para el ave, etc.). A su vez, los arcos también pueden representar ligazones muy diversas entre los nodos. Como se ve en la red semántica propuesta, es usual que aparezca la relación **«es»**: esta ligadura representa la relación de «instanciación» o «particularización» de un elemento de una clase. Mediante la ligazón «es», se suelen vincular los elementos a la «clase» de los elementos que los engloba; así, aparece que «avestruz es ave», igual que «loro es ave» y del mismo modo ave es una instanciación de una superclase que engloba a las aves y otros animales («ave es animal»).

También es usual que aparezca la relación **«tiene»**. Los elementos vinculados mediante la relación «tiene» son partes del elemento que aparece en el nodo ligado. Así, el avestruz tiene patas, tiene cuello, etc. Pero esta relación es distinta de la instanciación: las patas no son un caso, un ejemplo del avestruz; del mismo modo, un ave no está compuesta de avestruces, loros, etc.

Aparece numerosas veces también la relación **«es-de»**, que liga un objeto de un nodo con los nodos donde aparecen las «cualidades» que el objeto tiene: por ejemplo, «patas» es-de «tamaño» (grande), «cuello» es-de «longitud» (larga), etc.

Otra relación genérica es la relación **«valor»** que liga una cualidad con los posibles valores que adopta (así, el tamaño de las patas es grande, el color del loro verde, etc.).

Aparecen, por último, otro conjunto de ligazones diversas que representan otras propiedades o informaciones acerca de los elementos de la red: el ave «puede» volar, el «número de» patas es dos, etc.

Una de las mayores ventajas de la representación mediante redes semánticas es la enorme flexibilidad que aporta a la descripción del conocimiento. En efecto, por su propia naturaleza, es admisible ampliar en cualquier momento la red mediante la adición de nuevos nodos con sus arcos de ligadura correspondientes.

Otra de las grandes ventajas es la «herencia» o transmisión de cualidades a lo largo de la red.

Si en la red propuesta desarrollamos toda la información de que dispongamos sobre los «animales» o sobre las «aves» (como hemos hecho con el nodo «volar»), toda esa información queda disponible para ser aplicable a todas las instancias de la clase «ave» (en nuestro caso «loro», «avestruz», etc.), de tal modo que una vez definida la estructura básica de la red, la ampliación de conocimientos sobre una clase produce automáticamente la ampliación de la información sobre las instancias de esa clase, sin que tenga que ser repetida la información: se evita así mucha redundancia de conocimientos.

La herencia, por el contrario, tiene el inconveniente de que dificulta el establecimiento de excepciones (por ejemplo, si queremos representar el hecho de que el avestruz no vuela, a pesar de ser un ave).

Por otro lado, la definición de la red semántica supone una cierta estructuración del conocimiento, pero que «viene dada», no que es impuesta, desde el exterior a partir de un proceso de abstracción.

■ Ternas objetivo-atributo-valor

Una forma especial de red semántica aparece en el esquema de ternas o tripletas «objeto-atributo-valor». El conjunto de conocimientos de que disponemos se organiza en forma de una colección de ternas de elementos. En ellas el objeto puede ser un elemento físico o una entidad abstracta o conceptual; el atributo representa una cualidad o característica del objeto; el tercer elemento de la terna es el valor que, en un momento dado, toma el atributo que aparece en la terna, referido al objeto de que se trata.

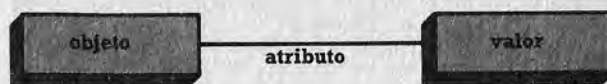


Fig. 2. Terna O-A-V.

Así, a partir de los conocimientos que representábamos en la red semántica descrita anteriormente, se podría establecer la serie de ternas que se incluyen en la figura.

TEMAS MONOGRAFICOS DE VANGUARDIA

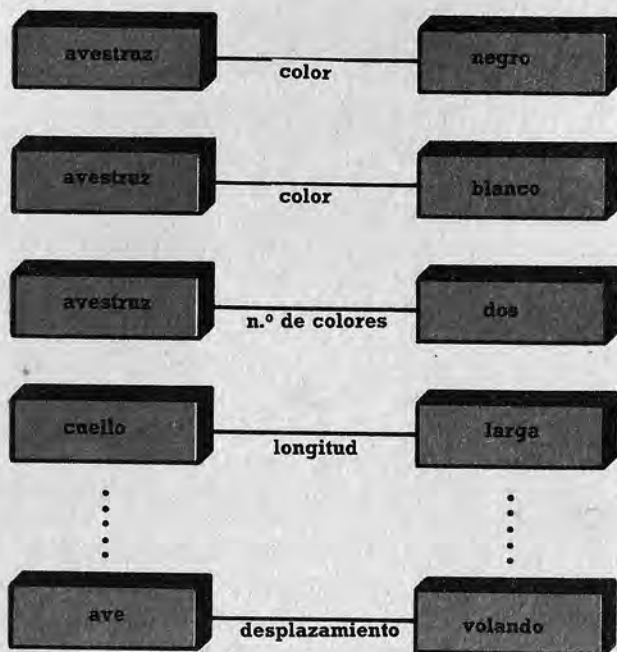


Fig. 3. Representación en ternas.

Este modo de representar los conocimientos es menos flexible (más estructurado), pero puede ser más claro. Por otro lado, normalmente será útil incluir alguna información adicional sobre cómo están relacionados los objetos entre sí. En el caso que estamos tomando de ejemplo se podría construir un árbol que relacionara los objetos de las ternas, tal como el que se muestra en la figura.

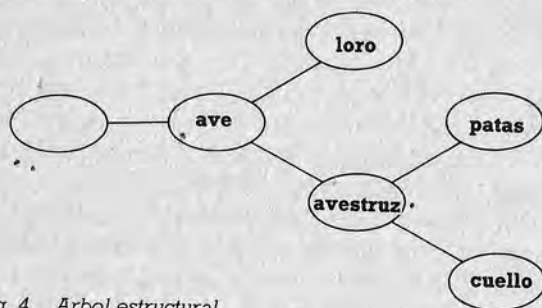


Fig. 4. Arbol estructural.

El vínculo (implícito) establecido entre el atributo y el objeto es de tipo «es-de», y el que existe entre el atributo y su valor es, precisamente, de tipo «valor». Las relaciones de instanciación (el ave puede ser un avestruz, un loro, un águila...) quedan, en principio, fuera de la tripleta y hay que establecerlas externamente. Lo mismo se puede decir de la relación de pertenencia o «ser-parte-de» (el avestruz tiene «patas», «cuello», etc.).

En ocasiones se puede establecer un esquema genérico de ternas en los que apa-

recen los objetos y sus atributos, pero sin asignación alguna de valores. Este grupo de ternas representaría el conjunto de informaciones que debe estar (o suele estar) disponible en el sistema, sin referirse a ningún caso concreto. Es la «infraestructura» o «estructura estática» de la base de conocimientos.

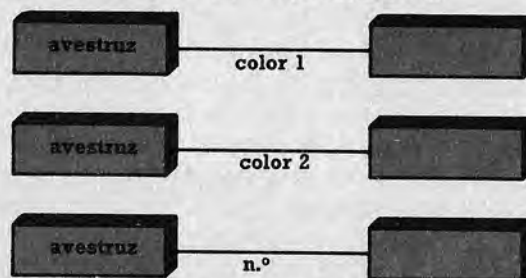


Fig. 5. Esqueleto de las ternas.

Cuando se «llenan» los espacios correspondientes de las ternas (cuando se asignan valores), se está incluyendo la parte «parte dinámica» de la base de conocimientos. Se opera, bajo este esquema, con la representación estática de la base y, en el momento oportuno, se manejan los valores dinámicos: se establece una primera asignación de valores, pero en el transcurso de la operación se obtienen o facilitan valores adicionales que enriquecen nuestra base de conocimientos.

Naturalmente si, se amplía el concepto de terna para representar no sólo los objetos, sino categorías superiores, la estructura se hace más flexible, pero puede ser más difícil su manejo.

Es usual, sin embargo, que aparezcan ternas con información de objetos a distintos niveles e incluso algunas correspondientes a categorías que engloban las anteriores: en el ejemplo anterior, aparte las ternas correspondientes al avestruz o al loro, aparecían valores para ciertos atributos del objeto «ave».

El tipo y el nivel de representación, utilizado vendrá dado, en multitud de ocasiones, por el entorno en que estemos moviéndonos y las herramientas de que dispongamos.

En los casos sencillos, los conocimientos de que dispongamos pueden referirse sólo a un objeto y, entonces, se suprime el elemento «objeto» de la terna, pasando ésta a ser, simplemente, una «pareja» de atributo valor.

Aunque parece una limitación poco interesante, puede ser útil en ocasiones por la simplicidad que introduce y el aumento de eficacia que aporta. De hecho existen sistemas funcionando contruidos bajo este esquema de representación.

TERMINOLOGIA

Adquisición de conocimientos. Proceso de extracción, estructuración y organización del conocimiento, partiendo de alguna fuente de conocimiento, generalmente una persona experta, y que puede utilizarse en un programa.

Backtracking. Técnica de búsqueda que va seleccionando en cada nodo del gráfico el nodo adyacente que se tomará a continuación. Si éste es un nodo que no satisface las condiciones que se están comprobando —«fallo»—, se tomará otro nodo adyacente al original.

Si todos los nodos adyacentes son «fallos», el propio nodo, considerado será «fallo». De esta forma, a través de la secuencia de inferencias se van probando los distintos caminos. En la planificación de programas, este sistema se utiliza para ir probando uno tras otro todos los planes, de forma que vayan identificándose los que sean inaceptables.

Base de conocimientos. Una parte del sistema basado en el conocimiento, o sistema experto que contiene el conocimiento sobre el dominio de que se trate.

Búsqueda en anchura. En una jerarquía de reglas u objetos; este tipo de búsqueda indica una estrategia en la que todas las reglas y objetos dentro del mismo nivel de jerarquía se examinan antes que cualquiera de las reglas u objetos del siguiente nivel inferior.

Búsqueda en profundidad. En un sistema jerárquico de reglas y objetos, la búsqueda en profundidad indica una estrategia en la cual se examina una regla y objeto del nivel más alto y a continuación las reglas y objetos situados inmediatamente debajo del

examinado. Al proceder de este modo, el sistema busca por una rama simple del árbol jerárquico, hasta terminarla. En contraposición a búsqueda en anchura.

Cadena de inferencia. Secuencia de pasos de reglas de aplicación utilizados por un sistema basado en reglas para llegar a las conclusiones.

Certeza. Grado de confianza en un hecho determinado o en una relación. Se representa por un número, dentro de una gama de valores, en el que el superior denota una certeza absoluta de que la aseveración es cierta, y el inferior la certeza también absoluta de que la aseveración es falsa. En IA este término se utiliza en contraposición a probabilidad, que indica las posibilidades existentes de que ocurra un determinado acontecimiento. Este término es sinónimo de factor de confianza.

Conocimientos de metanivel. Es el conocimiento sobre sí mismo que tiene un sistema de ordenador, la extensión y fiabilidad de sus conocimientos sobre el dominio, y cuánto y cómo mejorar sus conocimientos sobre el dominio.

Dominio. En matemáticas, conjunto de valores que debe asumir el argumento de una función. En sistemas expertos, el campo del conocimiento o clase de las tareas que puede abordar.

Espacio de un problema. Gráfico en el que los nodos representan todos los estados posibles de las soluciones parciales o completas del problema, y los arcos representan los operadores que transforman un estado en otro. La búsqueda de la solución al problema que se está considerando se representa estableciendo un camino a partir del

TERMINOLOGIA

nodo que representa el estado inicial, hasta el nodo que representa el estado objetivo.

Instanciación. Patrón o fórmula en la que se han sustituido las variables por constantes. En un sistema de producción, la instanciación es el resultado de confrontar una regla con el contenido de la memoria de datos. Puede presentarse en forma de par ordenado, en el que el primer miembro identifica la regla que se ha satisfecho, y el segundo es una lista de elementos de la memoria de trabajo que satisfacen los elementos condicionales de la regla.

Jerarquía. Conjunto ordenado de conceptos y objetos, en el que algunos están subordinados a los otros. Las jerarquías normalmente implican herencias, y así, los objetos o conceptos en que se encuentran más arriba en la organización «contienen» los objetos o conceptos que se encuentran debajo.

Lenguaje de ingeniería del conocimiento de propósito general. Lenguaje de ordenador diseñado para construir sistemas expertos e incorporarles características que los hacen aplicables a distintos problemas, áreas y tipos.

Mantenimiento de un sistema experto.

A diferencia del software de ordenador tradicional, que se actualiza con no demasiada frecuencia, los sistemas expertos, por su propia naturaleza, son muy sencillos de modificar.

La mayoría de los sistemas expertos que están funcionando se están mejorando constantemente, con la edición de nuevas reglas. En la mayoría de las aplicaciones, la organización del usuario suele desear establecer una rutina que capture e incorpore nuevos conocimientos al sistema. Puede ser muy útil responsabilizar a una persona para que introduzca nuevas reglas, siempre que los datos o procedimientos cambien, o siempre que aparezcan cuestiones que el sistema vigente no pueda resolver.

Memoria de datos. Base de datos global en un sistema de producción. El contenido puede ordenarse total o parcialmente en base a su tiempo de creación o modificación más reciente. La memoria de datos suele ser la parte más volátil del sistema de producción.

Memoria de producción. Conjunto de todas las reglas de un sistema de producción. Los restantes componentes del sistema de producción son la memoria de datos y la máquina de inferencias.

Memoria de reglas. Otra denominación de la memoria de producción.

Propagación de probabilidades. En una red de inferencias, ajuste de probabilidades en los nodos, para tomar nota del efecto de nueva información sobre la probabilidad de un nodo particular.

Red semántica. Estructura de datos para representar el conocimiento declarativo. Esta estructura es un gráfico en el que los nodos representan conceptos o elementos diversos y los arcos (que pueden llevar etiquetas) representan las relaciones entre los conceptos.

Representación de conocimientos. Dícese del proceso de construcción de sistemas expertos.

Sistema basado en el conocimiento. Programa en el cual el conocimiento del dominio es explícito y separado de otros conocimientos del programa.

Valor. Cantidad o cualidad que se puede utilizar para describir un atributo. Si consideramos, por ejemplo, el atributo «color», los posibles atributos de color son todos los nombres de los colores que conocemos. Si consideramos un objeto en particular, estaremos asignando un valor específico al atributo en cuanto indiquemos «El coche es rojo brillante».

Paralelo, almacenamiento. Dispositivo al que se puede acceder para obtener información en paralelo, es decir, de forma simultánea a varios. En contraposición a almacenamiento asociativo.

Paralelo conexión en. Transmisión simultánea en varios hilos o líneas de los bits de una palabra o dato. En contraposición a transmisión en serie.

Paralelo, impresora. (Ver **Impresora paralelo.**)

Paralelo, operación en. Ejecución de dos o más operaciones de forma concurrente o simultánea mediante varios dispositivos que pueden ser unidades lógicas, unidades aritméticas, etc.

Paralelo, ordenador. Ordenador que dispone de varias unidades aritméticas o lógicas que se utilizan para realizar operaciones en paralelo, o procesos en paralelo. En contraposición a ordenador escalar, en el que las operaciones se llevan a cabo de forma secuencial. Históricamente, ordenador que disponía de los recursos necesarios para manipular todos los bits de una palabra en paralelo.

Paralelo, proceso en. Ejecución simultánea de dos o más procesos en dispositivos múltiples, como pueden ser varios canales o unidades de proceso. En contraposición a operación en serie.

Paralelo, puerta en. Dispositivo que permite que los datos entren o salgan en paralelo, a través de varios hilos o canales de transmisión, en contraposición a puerta en serie.

Paralelo, transmisión. En comunicaciones, transmisión simultánea de cierto número de elementos de señal que constituyen la misma señal o dato. Se puede, por ejemplo, utilizar un código mediante el cual cada señal se caracteriza por una combinación de tres o más frecuencias simultáneas que se envían por el canal. En contraposición a transmisión en serie.

Parameter. (Véase **Parámetro.**)

Parámetro. Variable independiente que toma valores fijos. También puede referirse a algún dato solicitado por un programa o subrutina para condicionar su ejecución o seleccionar uno de los posibles modos de funcionamiento.

Parcheo. Cambios realizados en una rutina o procedimiento, eliminando o añadiendo restricciones sencillas para corregir errores. También se puede referir a modificar un programa de forma fácil.

Parenthesis free notation. Término inglés para indicar la notación con prefijos.

Paridad, comprobación de la. Comprobación para averiguar si el número de unos (o de ceros) de una matriz de dígitos binarios es par o impar.

Parity bit. (Véase **Bit de paridad.**)

Parity chek. Término inglés que significa comprobación de paridad. (Véase **Comprobación de paridad. Paridad, comprobación.**)

PASCAL. Lenguaje de programación de alto nivel escrito a principios de la década de los sesenta por Niklaus Wirth, cuya característica principal es la posibilidad de estructurar los programas.

Patch. Término inglés que indica parcheo, o corrección. Se refiere a los cambios realizados en rutinas o procedimientos, añadiendo o eliminando restricciones para corregir los errores.

Path. Término inglés que significa camino, recorrido. En informática indica un canal lógico por el que se transfieren los datos. También se utiliza para designar el «camino» que lleva desde el directorio raíz hasta la posición de un fichero o programa en una estructura de directorios y subdirectorios.

PCB. Siglas de Printed Circuit Board. Tarjeta de circuito impreso.

PCM. Pulse Code Modulation. Término inglés que significa modulación por codificación de impulsos. Codificación digital muy utilizada en comunicaciones. Cada carácter se codifica en varios impulsos, que son los que modulan otra señal llamada portadora.

Periférico. Dispositivo o equipo que se encuentra separado de la unidad central de proceso, pero que trabaja controlado por ella. Suelen ser un sistema de comunicación del sistema con el exterior, aunque pueden también recibir y tomar datos, para ir enviándolos a la unidad central, almacenar ficheros y otras muchas tareas. Entre los periféricos, podemos citar impresoras, terminales de pantalla, plotters, unidades de disco (flexibles y rígidos), etc.

Peripheral unit, o equipment. (Véase **Periférico**).

Pila. Conjunto de registros o posiciones de memoria que se organizan según una técnica especial llamada LIFO (last in first out el último en llegar, primero en salir). Esto significa que, como en una pila de leña, los últimos datos en haber sido escritos son los que se encuentran en la posición de «más arriba» en la pila y, por tanto, se pueden extraer los primeros.

Pista. Parte o zona de un disco donde se graban los datos, sin que se desplace la cabe-

za. Las pistas forman sectores circulares concéntricos. Los discos suelen tener cientos de pistas, en una o en ambas caras.

Pixel. Contracción de «picture element». Elemento de imagen. Es el elemento más pequeño de una imagen digitalizada. Cada uno de los puntos o elementos de la pantalla (la imagen no podría dividirse en elementos más pequeños).

PL/1. Siglas de Programming Language/1. Lenguaje de programación de alto nivel, creado para aprovechar las mejores características del COBOL (tan útil en problemas de gestión) y del FORTRAN (pensado para programas de tipo científico).

Planificación. Método para organizar varios trabajos o tareas o para responder adecuadamente a peticiones simultáneas.

Plotter. Periférico específico para dibujar gráficos. Existen modelos muy diferentes. Unos utilizan un lápiz y otros varios lápices (tres o cuatro). Se pueden seleccionar distintos colores y grosores para los trazos. En su estructura fundamental existen dos tipos, los analógicos (continuos) y los digitales (por pasos), dependiendo de las señales que reciben. Pueden trazar sobre un plano o sobre un cilindro. Los plotters pueden conectarse a un ordenador, para recibir las órdenes de trazado de los gráficos. Si el plotter es analógico, para poder conectarlo a un ordenador convencional será necesario utilizar un conversor analógico-digital.

Plug in. Término inglés que significa enchufable. Se refiere a que basta con introducir el conector en la toma adecuada. También puede referirse a la compatibilidad entre dos dispositivos. Si son plug in, son compatibles, y basta desconectar uno y conectar el otro para seguir trabajando como anteriormente.

Pocket computer. Término inglés para designar a las calculadoras sofisticadas de bolsillo, que utilizan tecnología de integración a gran escala.

